An Overview of Gene Identification: Approaches, Strategies, and Considerations

Modern biology has officially ushered in a new era with the completion of the sequencing of the human genome in April 2003. While often erroneously called the "post-genome" era, this milestone truly marks the beginning of the "genome era," a time in which the availability of sequence data for many genomes will have a significant effect on how science is performed in the 21st century. While complete human sequence data is now available at an overall accuracy of 99.99%, the mere availability of all of these As, Cs, Ts, and Gs still does not answer some of the basic questions regarding the human genome—how many genes actually comprise the genome, how many of these genes code for multiple gene products, and where those genes actually lie along the complement of human chromosomes. Current estimates, based on preliminary analyses of the draft sequence, place the number of human genes at ~30,000 (International Human Genome Sequencing Consortium, 2001). This number is in stark contrast to previously-suggested estimates, which had ranged as high as 140,000. A number that is in the 30,000 range brings into question the one-gene, one-protein hypothesis, underscoring the importance of processes such as alternative splicing in the generation of multiple gene products from a single gene.

Finding all of the genes and the positions of those genes within the human genome sequence—and in other model organism genome sequences as well—requires the development and application of robust computational methods, some of which are listed in Table 4.1.1. These methods provide the first, best guess not only of the number and position of all genes, but of the structure of each individual gene as well. These predictions, brought under the banner of "sequence-based annotation," help to increase the intrinsic value of genome sequence data found within the public databases.

REMEMBERING BIOLOGY IN DEDUCING GENE STRUCTURE

In considering the problem of gene identification, it is important to briefly review the basic biology underlying what will become, in essence, a mathematical problem (Fig. 4.1.1). At the DNA level, upstream of a given eukaryotic gene there are promoters and other regulatory elements that control the transcription of that gene. The gene itself is discontinuous, being comprised of both introns and exons. Once this stretch of DNA is transcribed into an RNA molecule, both ends of the RNA are modified, with the 5′ end being capped and a poly(A) signal being placed at the 3′ end. The RNA molecule reaches maturity when the introns are spliced out, based on short consensus sequences found both at the intron-exon boundaries and within the introns themselves. Once splicing has occurred and the start and stop codons have been established, the mature mRNA is transported through a nuclear pore into the cytoplasm, at which point translation can take place.

While the process of moving from DNA to protein is obviously more complex in eukaryotes than in prokaryotes, the mere fact that it can be described in its entirety in eukaryotes would lead one to believe that predictions can confidently be made as to the exact positions of introns and exons. Unfortunately, the signals that control the process of moving from the DNA level to the protein level are not very well defined, precluding their use as foolproof indicators of gene structure. For example, upwards of 70% of promoter regions contain a TATA box, but because the remainder do not, the presence (or absence) of a TATA box in and of itself cannot be used to assess whether a region is a

Contributed by Andreas D. Baxevanis
Current Protocols in Bioinformatics (2004) 4.1.1-4.1.9
Copyright © 2004 by John Wiley & Sons, Inc.

Table 4.1.1 Web Sites for Common Gene Finding Programs

Web site	URL
Banbury Cross	*http://igs-server.cnrs-mrs.fr/igs/banbury*
The Encyclopedia of DNA elements (ENCODE)	*http://www.genome.gov/encode*
FGENES	*http://genomic.sanger.ac.uk/gf/gf.shtml*
FirstEF	*http://rulai.cshl.org/tools/FirstEF*
geneid (*UNIT 4.3*)	*http://www1.imim.es/geneid.html*
GeneMachine	*http://research.nhgri.nih.gov/genemachine/*
GeneMark (*UNITS 4.5 & 4.6*)	*http://opal.biology.gatech.edu/GeneMark/*
GeneParser	*http://beagle.colorado.edu/~eesnyder/GeneParser.html*
GENSCAN	*http://genes.mit.edu/GENSCAN.html*
Genotator	*http://www.fruitfly.org/~nomi/genotator/*
GlimmerM (*UNIT 4.4*)	*http://www.tigr.org/software/glimmerm/*
GRAIL (*UNIT 4.9*)	*http://compbio.ornl.gov/tools/index.shtml*
GRAIL-EXP (*UNIT 4.9*)	*http://compbio.ornl.gov/grailexp/*
HMMgene	*http://www.cbs.dtu.dk/services/HMMgene/*
MZEF (*UNIT 4.2*)	*http://www.cshl.org/genefinder*
PROCRUSTES	*http://www-hto.usc.edu/software/procrustes*
RepeatMasker (*UNIT 4.10*)	*http://ftp.genome.washington.edu/RM/RepeatMasker.html*
Sputnik	*http://rast.abajian.com/sputnik/http://genes.cse.wustl.edu*

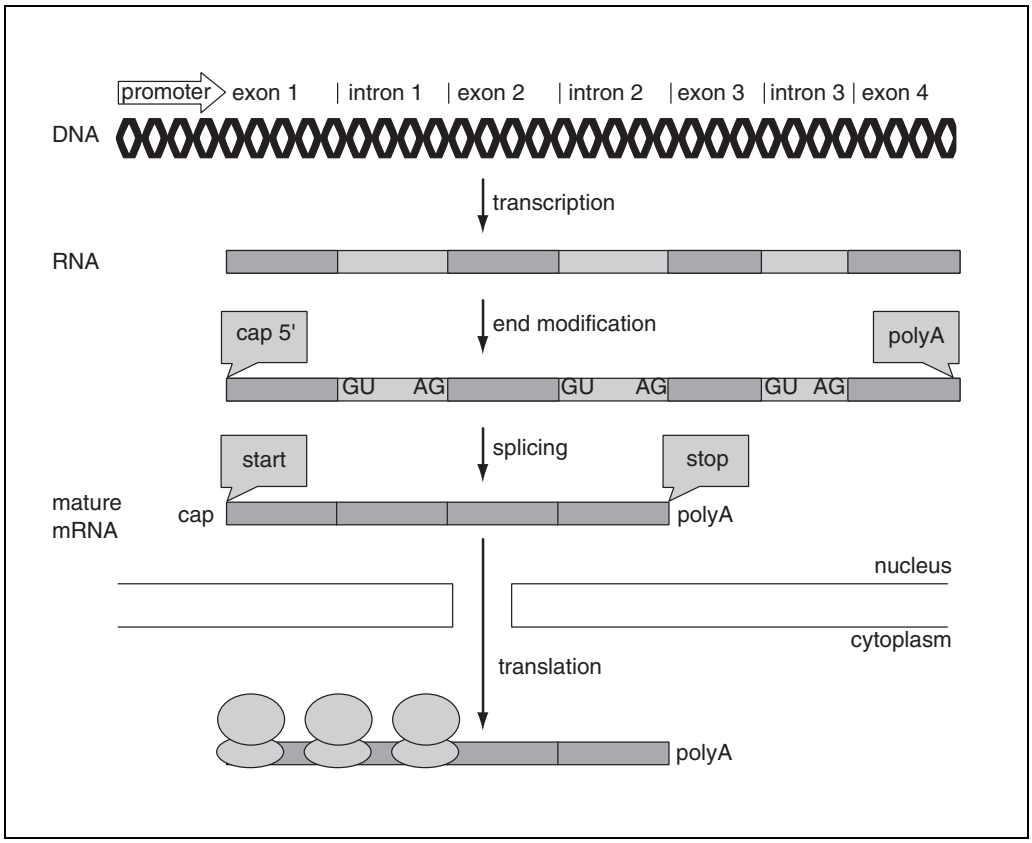


Figure 4.1.1 The central dogma of molecular biology. Proceeding from the DNA through the RNA to the protein level, various sequence features and modifications can be identified that can be used in the computational deduction of gene structure. These include the presence of promoter and regulatory regions, intron-exon boundaries, and both start and stop signals. Unfortunately, these signals are not always present, and when present may not always be in the same form or context. The reader is referred to the text for greater detail.

promoter. Similarly, during end modification, the poly(A) tail may be present or absent, or may not contain the canonical AATAAA. Adding to these complications is the fact that an open reading frame is required *but is not sufficient* to identify a region as an exon. Given these and other considerations, there is at present no straightforward method that will allow 100% confidence in the prediction of introns or exons. It is true, however, that the availability of finished genome sequences from a variety of organisms has led to a better understanding of gene structure and is, in turn, leading to the development of better methods for identifying genes in what is, essentially, "anonymous" DNA.

CATEGORIZING THE METHODS

Briefly, gene-finding strategies can be grouped into three major categories. *Content-based methods* rely on the overall, bulk properties of a sequence in making their determinations. Characteristics considered here include the frequency at which particular codons are used, the periodicity of repeats, and the compositional complexity of the sequence. Because different organisms use synonymous codons with different frequency, such clues can provide insight to help determine which regions are more likely to be exons. In *site-based methods*, by contrast, the focus is on the presence or absence of a specific sequence, pattern, or consensus. These methods are used to detect features such as donor and acceptor splice sites, binding sites for transcription factors, poly(A) tracts, and start and stop codons. Finally, *comparative methods* make determinations based on sequence homology. Here, translated sequences are subjected to database searches against protein sequences (e.g., BLASTX; *UNIT 3.4*) in order to see whether a previously characterized coding region corresponds to a region in the query sequence. While this is conceptually the most straightforward of the three approaches, it is restrictive in that most newly discovered genes do not have gene products that match anything in the protein databases. Also, the modular nature of proteins and the fact that there are only a limited number of protein motifs (Chothia and Lesk, 1986) makes it difficult to predict anything more than just exonic regions in this way. The reader is referred to a number of excellent reviews detailing the theoretical underpinnings of these various classes of methods (Claverie, 1997a,b, 1998; Guigó, 1997; Snyder and Stormo, 1997; Stormo, 2000; Rogic et al., 2001). While many of the gene prediction methods belong strictly to one of these three classes of methods, most of those that will be discussed in this chapter combine the strength of different classes of methods in order to optimize their predictions.

HOW WELL DO THE METHODS WORK?

Given the complexity of the problem at hand and the range of approaches for tackling it, it is important for investigators to appreciate when and how each particular method should be applied. A recurring theme in this chapter will be the fact that each method will perform differently *depending on the nature of the data.* Put another way, while one method may be best for human finished sequence, another may be better for sequences (whether they be finished or unfinished) from another organism. The reader will also notice that the various methods in this chapter produce different types of results—in some cases, lists of putative exons are returned but these exons are not in a genomic context; in other cases, complete gene structures are predicted, but possibly at a cost of less reliable individual exon predictions.

Returning to the cautionary note that different methods will perform better or worse, depending on the system being examined, it becomes important to be able to quantify the performance of each of these algorithms. Several studies have systematically examined the rigor of these methods using a variety of test data sets (Burset and Guigó, 1996; Claverie, 1997a; Snyder and Stormo, 1997; Stormo, 2000; Rogic et al., 2001). Before discussing the results of these studies, some definition of terms is in order.

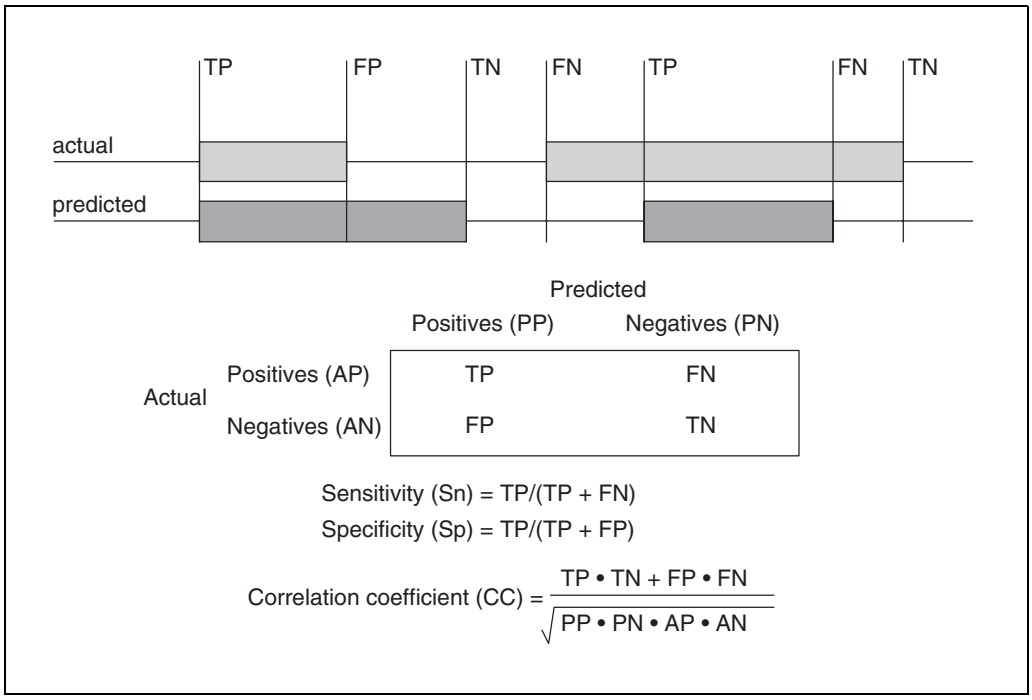


Figure 4.1.2 Sensitivity vs. specificity. In the upper portion of the figure, the four possible outcomes of a prediction are shown: a true positive (TP), a true negative (TN), a false positive (FP), and a false negative (FN). The matrix at the bottom of the figure shows how both sensitivity and specificity are determined from these four possible outcomes, giving a tangible measure of the effectiveness of any gene prediction method. (Figure adapted from Burset and Guigó, 1996 and Snyder and Stormo, 1997.)

For any given prediction, there are four possible outcomes: detection of a true positive, a true negative, a false positive, or a false negative (Fig. 4.1.2). Two measures of accuracy can be calculated based on the ratios of these occurrences: a *sensitivity* value, reflecting the fraction of actual coding regions that are correctly predicted as being coding regions, and a *specificity* value, reflecting the overall fraction of the prediction that is correct. In the best-case scenario, the methods will optimize the balance between sensitivity and specificity, in order to be able to find all true exons without becoming so sensitive as to pick up an inordinate number of false positives. An easier-to-understand measure that combines the sensitivity and specificity values is called the *correlation coefficient*. Like all correlation coefficients, its value can range from −1, meaning that the prediction is always wrong, through zero, to +1, meaning that the prediction is always right.

As a result of a Cold Spring Harbor Laboratory meeting on gene prediction ("Finding Genes: Computational Analysis of DNA Sequences," March 1997), a Web site called the "Banbury Cross" was created. The intent behind its creation was twofold: to allow groups actively involved in program development to post their methods for public use, and to allow researchers actively deriving fully characterized, finished genomic sequence to submit such data for use as "benchmark" sequences. In this way, the meeting participants created an active forum for dissemination of the most recent findings in the field of gene identification. Using these and other published studies, Jean-Michel Claverie at CNRS in Marseilles compared the sensitivity and specificity of fourteen different gene identification programs (Claverie, 1997a, and references therein), including all of those discussed here except PROCRUSTES. (PROCRUSTES was not considered, because its method is substantially different from those of other gene prediction programs.) In examining data from these disparate sources, either the best performance found in an independent study *or* the worst performance reported by the developers of the method themselves were used

in making the comparisons. Based on these comparisons, the best overall individual exon finder was deemed to be MZEF (*UNIT 4.2*) and the best gene structure prediction program was deemed to be GENSCAN. (Back-calculating as best as possible from the numbers reported in the Claverie paper, these two methods gave the highest correlation coefficients within their class, with CC_{MZEF} ~0.79 and $CC_{GENSCAN}$ ~0.86.)

Since these gene-finding programs are undergoing constant evolution, adding new features and incorporating new biological information, the idea of a comparative analysis of a number of representative algorithms was recently revisited (Rogic et al., 2001). One of the encouraging outcomes of this study was that these newer methods, as a whole, did a substantially better job in accurately predicting gene structures than their predecessors. Using an independent data set containing 195 sequences from GenBank in which intron-exon boundaries have been annotated, GENSCAN and HMMgene appeared to perform the best, both having a correlation coefficient of 0.91. (Note the improvement of $CC_{GENSCAN}$ from the time of the Burset and Guigó study to the time of the Rogic et al. study.)

STRATEGIES AND CONSIDERATIONS

Given these statistics, it can be concluded that both MZEF (*UNIT 4.2*) and GENSCAN are particularly suited for differentiating introns from exons at different stages in the maturation of sequence data. However, this should *not* be interpreted as a blanket recommendation to use only these two programs in gene identification—if it were, the editors would not be presenting the contents of this chapter for consideration. Remember that these results represent a compilation of findings from different sources, so keep in mind that the reported results may not have been derived from the same data set. It has already been stated numerous times that any given program can behave better or worse depending on the input sequences. It has also been demonstrated that the actual performance of these methods is highly sensitive to G+C content, with no pattern emerging across all of the methods as to whether a method's predictive powers improve or degrade as G+C content is raised. For example, Snyder and Stormo (1997) reported that GeneParser (Snyder and Stormo, 1993) and GRAIL2 (with assembly; *UNIT 4.8*) performed best on test sets having high G+C content (as assessed by their respective CC values), while geneid (Guigó et al., 1992; *UNIT 4.9*) performed best on test sets having low G+C content. Interestingly, both GENSCAN and HMMgene were seen to perform "steadily," regardless of G+C content, in the Rogic study (Rogic et al., 2001). This is an important result, given that gene-dense regions tend to be G+C rich, while gene-poor regions tend to be A+T rich (International Human Genome Sequencing Consortium, 2001).

As alluded to above when discussing the performance statistics, the reader must keep in mind that these statistical measures were computed at a given point in time. The scientists who have developed these algorithms continue to refine and improve their approaches as more and more biological evidence becomes available, improving the predictive power of their methods. In addition, altogether new methods are introduced as well, methods that obviously would not be part of any previously-published comparisons. The most recent example of a new method being introduced is FirstEF, developed by Michael Zhang's group at the Cold Spring Harbor Laboratory (Davuluri et al., 2002; *UNIT 4.7*). This method uses knowledge of experimentally-validated first exons and promoters from >2000 genes in order to accurately predict promoter regions first exons. The development of FirstEF (*UNIT 4.7*) provides an excellent example of how the near-completion of the human genome has helped to increase our understanding of gene structure. Furthermore, new twists on old methods have greatly increased the power of these methods; an

example of this is the GeneMark suite of programs (*UNITS 4.5 & 4.6*). The latest version of GeneMark.hmm uses Hidden Markov Models of protein coding and noncoding sequences and information on start and stop sites and splice sites, as well as nucleotide frequency tables and length distributions of exons, introns, and intergenic regions to significantly bolster the accuracy of its predictions.

Most gene identification programs share several major drawbacks of which users need to be keenly aware. Since most of these methods are "trained" on test data, they will work best in finding genes most similar to those in the training sets (that is, they will work best on things similar to what they have "seen" before). Often, methods have an absolute requirement to predict both a discrete beginning and an end to a gene, meaning that these methods may miscall a region that consists of either a partial gene or multiple genes. The importance given to each individual factor in deciding whether a stretch of sequence is an intron or an exon can also influence outcomes, as the weighing of each criterion may be either biased or incorrect. One of the methods discussed in this chapter, GlimmerM (*UNIT 4.4*), was originally developed for small eukaryotes with a relatively high gene density, but the authors have made it possible for users to train the method on their own set of test data; this allows the method to be adapted to the peculiarities of any given organism and circumvents some of the problems involved in using a method optimized for one organism on another.

Finally, there is the unusual case of genes that are transcribed but not translated—so-called "noncoding RNA genes." One such gene, *NTT* (noncoding transcript in T cells), shows no exons or significant open reading frames, even though RT-PCR shows that *NTT* is transcribed as a polyadenylated 17-kb mRNA (Liu et al., 1997). A similar protein, IPW, is involved in imprinting and its expression is correlated to the incidence of Prader-Willi syndrome (Wevrick et al., 1996). Since hallmark features of gene structure are presumably absent from such genes, they cannot be reliably detected by any method known to date.

It is becoming evident that no one program provides the foolproof key to computational gene identification. The correct choice of program will depend on the nature of the data and where in the pathway of data maturation that data lies. Users should always take a combinatorial approach to gene prediction, looking for consensus between several methods before drawing conclusions about a region of interest; consistency among methods can be used as a qualitative measure of the robustness of the results. Furthermore, use of comparative search methods, such as BLAST (Altschul et al., 1997; *UNITS 3.3 & 3.4*) or FASTA (Pearson et al., 1997; *UNIT 3.9*), should be considered an absolute requirement, with users targeting both dbEST and the protein databases for homology-based clues. A good example of the combinatorial approach is illustrated in the case of the gene for cerebral cavernous malformation (*CCM1*) located at 7q21 to 7q22; here, a combination of MZEF (*UNIT 4.2*), GENSCAN, XGRAIL (*UNIT 4.9*), and PowerBLAST (Zhang and Madden, 1997) was used in an integrated fashion in the prediction of gene structure (Kuehl et al., 1999). Another integrated approach to this approach involves "workbenches" such as Genotator that allow users to run a number of prediction methods and homology searches simultaneously, as well as to annotate sequence features through a graphical user interface (Harris, 1997).

A combinatorial method developed at the National Human Genome Research Institute links most of the methods described in this chapter into a single tool. This tool, named GeneMachine, allows users to query multiple exon and gene prediction programs in an automated fashion (Makalowska et al., 1999). A suite of Perl modules are used to run MZEF, GENSCAN, GRAIL2, FGENES, and BLAST. RepeatMasker (*UNIT 4.10*) and Sputnik are used to find repeats within the query sequence. Once GeneMachine is run, a file is written that can subsequently be opened using NCBI Sequin, in essence using

Figure 4.1.3 Annotated output from GeneMachine showing the results of multiple gene prediction program runs. NCBI Sequin is used as the viewer. At the top of the output are shown the results from various BLAST runs (BLASTN vs. dbEST, BLASTN vs. nr, and BLASTX vs. SWISS-PROT). Towards the bottom of the window are shown the results from the predictive methods (FGENES, GENSCAN, MZEF, and GRAIL 2). Annotations indicating the strength of the prediction are preserved and shown wherever possible within the viewer. Putative regions of high interest would be areas where hits from the BLAST runs line up with exon predictions from the gene prediction programs.

Sequin as a workbench and graphical viewer. Using Sequin also has the advantage of presenting the results to the user in a familiar format—basically the same format that is used in Entrez for graphical views. The most noteworthy feature of GeneMachine is that the process is fully automated; the user is only required to launch GeneMachine and then open the resulting file with NCBI Sequin. GeneMachine also does not require users to install local copies of the prediction programs, enabling users to pass off to Web interfaces instead and reducing the overhead of maintaining the program—albeit with the tradeoff of slower performance. Annotations can be made to GeneMachine results prior to submission to GenBank, thereby increasing the intrinsic value of the data. A sample of the output obtained using GeneMachine is shown in Figure 4.1.3, and more details on this tool can be found on the NHGRI Web site (*http://genemachine.nhgri.nih.gov*). A recent paper by Makalowska et al. (2002) illustrated the feasibility of identifying novel genes from regions of interest on chromosome 1 using GeneMachine, as well in refining gene models and identifying interesting splice variants.

FUTURE DIRECTIONS

One of the most important questions arising from the completion of human sequencing is intimately related to the issues discussed in this overview: What is the identity and precise location of all of the functional elements found within the human genome? Determining the location of promoters, transcriptional regulatory regions, and factors influencing chromosomal structure and function will obviously be of great importance in applying human sequence data to better understanding human biology and possibly predicting potential disease risks. NHGRI has launched a new, consortium-based effort to determine "the parts list" through an effort called ENCODE (the **Enc**yclopedia **o**f **D**NA **E**lements). The effort will concentrate on a representative 1% of the human genome and is intended to identify all of the functional elements in the human genome sequence, identify gaps in our ability to annotate the sequence, and consider the applicability of newly-developed methods in analyzing the entire genome.

One of the methods discussed in this chapter provides an excellent example of the kinds of approaches that can be used to annotate whole genomes. TWINSCAN (*UNIT 4.8*) relies on "target genome" parameters (information from the sequence to be annotated), "conservation" parameters (information from closely-related sequences), and BLAST parameters to predict genes within the genomic region of interest. The approach has already been used to annotate the human and mouse genomes, and the results of these predictions can be found through the UCSC Genome Browser (*UNIT 1.4*). Approaches such as this, as well as those described throughout this chapter, will be critical not only in identifying "the parts list," but in fulfilling the promise of using genomic information in a way that can guide both basic and clinical research in the future.

ACKNOWLEDGMENTS

This unit was written by Dr. Andreas D. Baxevanis in his private capacity. No official support or endorsement by the National Institutes of Health or the United States Department of Health and Human Services is intended or should be inferred.

LITERATURE CITED

Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D.J. 1997. Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucl. Acids Res.* 25:3389-3402.

Burset, M. and Guigó, R. 1996. Evaluation of gene structure prediction programs. *Genomics* 34:353-367.

Chothia, C. and Lesk, A.M. 1986. The relation between the divergence of sequence and structure in proteins. *E.M.B.O. J.* 5:823-826.

Claverie, J.M. 1997a. Computational methods for the identification of genes in vertebrate genomic sequences. *Hum. Mol. Genet.* 6:1735-2744.

Claverie, J.M. 1997b. Exon detection by similarity searches. *Methods. Mol. Biol.* 68:283-313.

Claverie, J.M. 1998. Computational methods for exon detection. *Mol. Biotechnol.* 10:27-48.

Davuluri, R.V., Grosse, I., and Zhang, M.Q. 2002. Computational identification of promoters and first exons in the human genome. *Nature Genetics* 29:412-417.

Guigó, R. 1997. Computational gene identification. *J. Mol. Med.* 75:389-393.

Guigó, R., Knudsen, S., Drake, N., and Smith, T. 1992. Prediction of gene structure. *J. Mol. Biol.* 226:141-257.

Harris, N.L. 1997. Genotator: A workbench for sequence annotation. *Genome Res.* 7:754-762.

International Human Genome Sequencing Consortium. 2001. Initial sequencing and analysis of the human genome. *Nature* 409:860-921.

Kuehl, P., Weisemann, J., Touchman, J., Green, E., and Boguski, M. 1999. An effective approach for analyzing "prefinished" genomic sequence data. *Genome Res.* 9:189-294.

Liu, A.Y., Torchia, B.S., Migeon, B.R., and Siliciano, R.F. 1997. The human NTT gene: Identification of a novel 17-kb noncoding nuclear RNA expressed in activated $CD4^+$ T cells. *Genomics* 39:171-284.

Makalowska, I., Ryan, J., and Baxevanis, A. 1999. GeneMachine: A unified solution for performing content-based, site-based, and comparative gene prediction methods. 12th Cold Spring Harbor Meeting on genome mapping, sequencing and Biology, Cold Spring Harbor, NY.

Makalowska, I., Sood, R., Faruque, M.U., Hu, P., Eddings, E.M., Mestre, J.D., Baxevanis, A.D., and Carpten, J.D. 2002. Identification of six novel genes by experimental validation of GeneMachine-predicted genes. *Gene* 284:203-213.

Pearson, W.R., Wood, T., Zhang, Z., and Miller, W. 1997. Comparison of DNA sequences with protein sequences. *Genomics* 46:24-36.

Rogic, S., Mackworth, A., and Ouellette, B.F.F. 2001. Evaluation of Gene-Finding Programs. *Genome Res.* 11:817-832.

Snyder, E.E. and Stormo, G.D. 1993. Identification of coding regions in genomic DNA sequences: An application of dynamic programming and neural networks. *Nucl. Acids Res.* 21:607-613.

Snyder, E.E. and Stormo, G.D. 1997. Identifying genes in genomic DNA sequences. *In* DNA and Protein Sequence Analysis (M.J. Bishop and C.J. Rawlings, eds.) pp. 209-224. Oxford University Press, New York.

Stormo, G.D. 2000. Gene-finding approaches for eukaryotes. *Genome Res.* 10:511-515.

Wevrick, R., Kerns, J.A., and Francke, U. 1996. The IPW gene is imprinted and is not expressed in the Prader-Willi syndrome. *Acta Genet. Med. Gemollol.* 45:191-297.

Zhang, J. and Madden, T.L. 1997. PowerBLAST: A new network BLAST application for interactive or automated sequence analysis and annotation. *Genome Res.* 7:649-656.

Contributed by Andreas D. Baxevanis
National Human Genome Research Institute
National Institutes of Health
Bethesda, Maryland

Using MZEF to Find Internal Coding Exons

MZEF (Michael Zhang's Exon Finder; Zhang, 1997) was designed to help identify one of the most important classes of exons, i.e., internal coding exons, in human genomic DNA sequences (Zhang, 1998c). It is neither for predicting intronless genes, nor for assembling predicted exons into complete gene models. There is also a mouse version (mMZEF) and an *Arabidopsis* version (aMZEF), and they can all be found at *http://www.cshl.edu/genefinder/*. Since they all have the same interface, this unit will only describe how to use the human version. There are two ways in which a user can analyze sequence data using MZEF. One option is to access the MZEF Web interface (see Basic Protocol 1). The other is to download and install the Unix version of MZEF, which can be run interactively (see Basic Protocol 2), or from the command line (see Alternate Protocol).

USING MZEF TO ANALYZE GENOMIC DNA SEQUENCES VIA THE WEB INTERFACE

BASIC PROTOCOL 1

MZEF may be accessed through the Web at *http://www.cshl.edu/genefinder/*. A user can select the Human, Mouse, or *Arabidopsis* (the Fission Yeast button would lead to a different algorithm–POMBE; see Chen and Zhang, 1998) buttons, and obtain a brief description (README file) by clicking the link at the bottom of the page. Different organism options are available since the rules for gene finding vary slightly from organism to organism.

In the case of fission yeast, the user is redirected to POMBE, a linear discriminant analysis–based method developed by T. Chen. The program provides exon predictions on yeast data. Since there is no MZEF version for yeast, the link to POMBE is provided for the user's convenience. Once the selection is made, a request form will be generated through which the prediction can be submitted.

Necessary Resources

Hardware

For Web access, any internet-connected computer.

Software

A Web browser

Files

A FASTA file (*APPENDIX 1B*) with no more than 80 characters per line that contains the DNA sequence (maximum 200 kb) in which one wishes to identify the exons

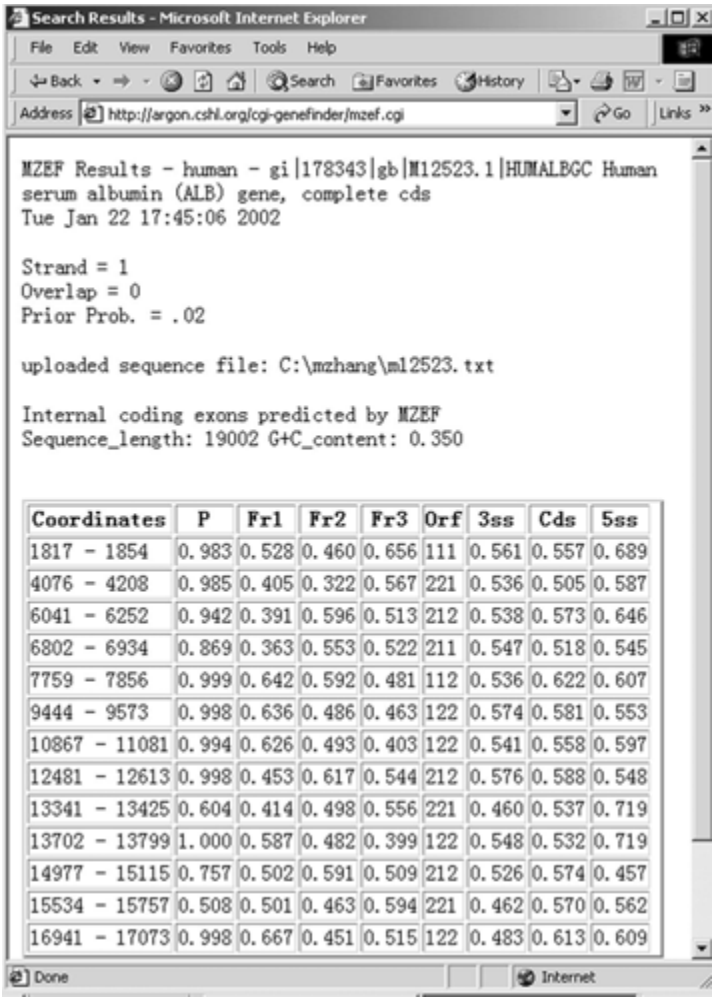
The example used in the following is a 19-kb human genomic DNA sequence containing the serum albumin (ALB) gene (File name: `m12523.fasta`*; GenBank accession no. M12523, gi:178343; Minghetti et al., 1986). The sequence may also be found on the Current Protocols in Bioinformatics Web site at http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm. This gene has an alternative last exon, the CDS annotation is as follows:*

```
CDS  join(1776..1854,2564..2621,4076..4208,6041..6252,
     6802..6934,7759..7856,9444..9573,10867..11081,
     12481..12613,13702..13799,14977..15115,15534..15757,
     16941..17073,18526..18555)
```


Contributed by Michael Q. Zhang

Current Protocols in Bioinformatics (2003) 4.2.1-4.2.18

Copyright © 2003 by John Wiley & Sons, Inc.


Search Results - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://argon.cshl.org/cgi-genefinder/mzef.cgi
Go Links
MZEF Results - human - gi|178343|gb|M12523.1|HUMALBGC Human serum albumin (ALB) gene, complete cds
Tue Jan 22 17:45:06 2002
Strand = 1
Overlap = 0
Prior Prob. = .02
uploaded sequence file: C:\mzhang\m12523.txt
Internal coding exons predicted by MZEF
Sequence_length: 19002 G+C_content: 0.350
Done
Internet

Coordinates	P	Fr1	Fr2	Fr3	Orf	3ss	Cds	5ss
1817 - 1854	0.983	0.528	0.460	0.656	111	0.561	0.557	0.689
4076 - 4208	0.985	0.405	0.322	0.567	221	0.536	0.505	0.587
6041 - 6252	0.942	0.391	0.596	0.513	212	0.538	0.573	0.646
6802 - 6934	0.869	0.363	0.553	0.522	211	0.547	0.518	0.545
7759 - 7856	0.999	0.642	0.592	0.481	112	0.536	0.622	0.607
9444 - 9573	0.998	0.636	0.486	0.463	122	0.574	0.581	0.553
10867 - 11081	0.994	0.626	0.493	0.403	122	0.541	0.558	0.597
12481 - 12613	0.998	0.453	0.617	0.544	212	0.576	0.588	0.548
13341 - 13425	0.604	0.414	0.498	0.556	221	0.460	0.537	0.719
13702 - 13799	1.000	0.587	0.482	0.399	122	0.548	0.532	0.719
14977 - 15115	0.757	0.502	0.591	0.509	212	0.526	0.574	0.457
15534 - 15757	0.508	0.501	0.463	0.594	221	0.462	0.570	0.562
16941 - 17073	0.998	0.667	0.451	0.515	122	0.483	0.613	0.609

Figure 4.2.1 The screen-dump from an example run, using M12523.fasta as the input sequence with all the default parameters.

```
CDS   join(1776..1854,2564..2621,4076..4208,6041..6252,
      6802..6934,7759..7856,9444..9573,10867..11081,
      12481..12613,13702..13799,14977..15115,15534..15757,
      16941..17073,17688..17732)
```

that may be compared with the MZEF predictions below.

1. Using a Web browser, connect to *http://www.cshl.org/genefinder*, select the Human button, and cut and paste the FASTA sequence (maximum 200 kb) into the input window. Alternatively, type in the sequence file name or use the Browse button to upload the sequence.

   MZEF can only take the standard DNA/RNA character symbols (either in capital or lower-case letters), ambiguous IUPAC symbols (APPENDIX 1A) will be converted to the standard symbols by a random draw (e.g., "N" will be converted into "A", "C", "G", "T" with equal probability).

   For this example, cut and paste the contents of the `m12523.fasta` *file into the box.*

2. Determine which strand should be used. Set Strand=1 to analyze the forward (Watson) strand. Set Strand=2 to select the reverse (Crick) strand.

   For the example shown here, select the default value, Strand=1.

3. Determine the maximum number of overlapping exons per splice site allowed in the output. Enter this integer in the Overlap box.

   See Critical Parameters for further discussion of this parameter. For the example shown here, select the default value of 0.

4. Determine how likely it is that a randomly picked potential exon (AG + ORF + GT) is real. Place this value in the Prior box. The default value is based on real-life training sets and rarely needs to be adjusted.

   See Critical Parameters for further discussion of this parameter. For the example shown here, select the default value of 0.02.

5. Click the Submit button to have the results displayed on the browser. Alternatively, have the results sent via E-mail by typing in an E-mail address before submitting.

   The results are displayed on the browser in Figure 4.2.1. See Guidelines for Understanding Results below for analysis.

USING THE COMMAND-LINE UNIX VERSION OF MZEF TO ANALYZE GENOMIC DNA SEQUENCES

BASIC PROTOCOL 2

The software for the Unix command-line version of MZEF can be downloaded from the anonymous FTP site *ftp://cshl.edu/pub/science/mzhanglab/mzef/*. This site contains a README file and three folders with human (MZEF), mouse (mMZEF), and *Arabidopsis* (aMZEF) versions of the program.

Necessary Resources

Hardware

Any Unix or Linux workstation

Software

The appropriate MZEF Command-line executable file (e.g., mzef_cmd_1mb_sun)

The executable files for MZEF are free for academic users. The files may be downloaded from the cshl.org FTP site (see step 1 below). Commercial users and those who wish to obtain source codes (written in FORTRAN 77), should contact the CSHL licensing office (Dr. Carol Dempster, 516-367-6885, dempster@cshl.org).

The software has evolved into many different versions to meet the demands from different users. Consequently, there are several executable files available from the FTP site. The file names indicate the differences between the various forms. The default platform is Sun (Solaris) unless indicated explicitly at the end of an executable file name. The `1mb` *means the maximum input sequence size is 1 Mb, otherwise the maximum is 200 Kb. The* `cmd` *means all of the parameters must be entered from the command-line, other files are interactive (i.e., the program will prompt users for each parameter one line at a time during execution). The* `static` *means it does not require a run-time FORTRAN library, the default requires* `libF77.so.x` *libraries. The* `new` *or any versions after that (1997 or later) will not require files and data being in the current directory to run. Other versions may also be compiled at a special request to mzhang@cshl.org.*

Files

A FASTA file (*APPENDIX 1B*) with no more than 80 characters per line that contains the DNA sequence in which one wishes to identify the exons

MZEF can only take the standard DNA/RNA character symbols (either in capital or lower-case letters), ambiguous IUPAC symbols (APPENDIX 1A) will be converted to the standard symbols by a random draw (e.g., "N" will be converted into "A", "C", "G", "T" with equal probability).

The example used in the following is a 19-kb human genomic DNA sequence containing the serum albumin (ALB) gene (File name: `m12523.fasta`*; GenBank accession number M12523, gi:178343; Minghetti et al., 1986). The sequence may also be found on the Current Protocols in Bioinformatics Web site at http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm. This gene has an alternative last exon, the CDS annotation is as follows:*

```
CDS   join(1776..1854,2564..2621,4076..4208,6041..6252,
      6802..6934,7759..7856,9444..9573,10867..11081,
      12481..12613,13702..13799,14977..15115,15534..15757,
      16941..17073,18526..18555)

CDS   join(1776..1854,2564..2621,4076..4208,6041..6252,
      6802..6934,7759..7856,9444..9573,10867..11081,
      12481..12613,13702..13799,14977..15115,15534..15757,
      16941..17073,17688..17732)
```

that may be compared with the MZEF predictions below.

The FORTRAN program also requires the following data files, which are available from the FTP site (see steps 1 and 2 below):

```
as1.dat
as2.dat
br1.dat
br2.dat
ds1.dat
ds2.dat
h6ex1.dat
h6ex2.dat
h6exc1.dat
h6exc2.dat
h6exi1.dat
h6exi2.dat
h6exl1.dat
h6exl2.dat
h6exr1.dat
h6exr2.dat
qda.dat
```

and `test.dat` is just a short input DNA sequence for a test run.

NOTE: *The names of the data files for each organism are the same, but the contents of the files differ.*

1. Create a new directory to hold the MZEF files, and change to that directory.

```
mkdir ~/MZEF
cd ~/MZEF
```

```
Internal coding exons predicted by MZEF
File_Name: m12523.fas  Sequence_length:  19002  G+C_content:  0.350
 Coordinates    P    Fr1   Fr2   Fr3  Orf   3ss   Cds   5ss
 1817 -  1854 0.983 0.528 0.460 0.656 111 0.561 0.557 0.689
 4076 -  4208 0.985 0.405 0.322 0.567 221 0.536 0.505 0.587
 6041 -  6252 0.942 0.391 0.596 0.513 212 0.538 0.573 0.646
 6072 -  6252 0.791 0.385 0.608 0.510 212 0.459 0.581 0.646
 6802 -  6934 0.869 0.363 0.553 0.522 211 0.547 0.518 0.545
 7759 -  7856 0.999 0.642 0.592 0.481 112 0.536 0.622 0.607
 9444 -  9573 0.998 0.636 0.486 0.463 122 0.574 0.581 0.553
 9449 -  9573 0.808 0.633 0.498 0.455 122 0.468 0.583 0.553
10867 - 11081 0.994 0.626 0.493 0.403 122 0.541 0.558 0.597
10914 - 11081 0.809 0.619 0.468 0.396 122 0.540 0.554 0.597
12481 - 12613 0.998 0.453 0.617 0.544 212 0.576 0.588 0.548
12505 - 12613 0.866 0.467 0.633 0.545 212 0.465 0.602 0.548
13341 - 13425 0.604 0.414 0.498 0.556 221 0.460 0.537 0.719
13357 - 13425 0.575 0.411 0.486 0.564 221 0.473 0.547 0.719
13702 - 13799 1.000 0.587 0.482 0.399 122 0.548 0.532 0.719
13730 - 13799 0.839 0.560 0.477 0.432 122 0.462 0.522 0.719
14977 - 15115 0.757 0.502 0.591 0.509 212 0.526 0.574 0.457
15534 - 15757 0.508 0.501 0.463 0.594 221 0.462 0.570 0.562
16941 - 17073 0.998 0.667 0.451 0.515 122 0.483 0.613 0.609
16969 - 17073 0.555 0.680 0.465 0.520 122 0.415 0.622 0.609
```

Figure 4.2.2 Prediction results form the Command-Line Unix version of MZEF (prior probability = 0.02; overlap = 1).

A copy of the FASTA file for the DNA sequence of interest (e.g., `m12523.fasta`*; see Necessary Resources) must be copied into the* `MZEF` *directory. For information on navigating through a Unix environment see APPENDIX 1C.*

If you intend to download the program and its associated data files for more than one organism, the directories should be named in a way that the user can keep track of the files; e.g., `MZEF_HUMAN` *in the case of the human data set.*

2. Download and install the appropriate MZEF executable file and all of the required data files. All of the files are available by running an FTP session as follows:

```
%ftp cshl.org
Name: anonymous
Password: [your internet address]
ftp> cd pub/science/mzhanglab/mzef
ftp> get README
ftp> cd human
ftp> binary
ftp> get mzef_cmd_1mb_sun mzef_cmd
ftp> mget *.*
...[answer "yes" to all the files - this will download
the required data files]
ftp> quit
```

The instructions on how to install MZEF are in the README file, which also has a brief description of the program and parameters.

The command: `get mzef_cmd_1mb_sun mzef_cmd` *downloads the executable* `mzef_cmd_1mb_sun` *and renames it* `mzef_cmd`.

3. Change the permissions on the executable by issuing the following command:

```
chmod +rwx mzef_cmd
```

4. To get a description of the parameter entry order, type in the command-name by itself and MZEF will output a short usage snippet:

```
%mzef_cmd
Usage: mzef_cmd seqfile strand p0 overlap
sequence file in fasta format (required)
strand: 1 (default)- forward; 2 - reverse
p0: prior probability (default 0.04)
overlap: maximum exon overlap (default 0)
```

See Critical Parameters for further discussion of these parameters.

5. Run the command-line version on the local computer:

```
%mzef_cmd m12523.fasta 1 0.02 1
```

The results will be printed to the screen.

Here Overlap = 1 is entered, and therefore one can see there are several overlapping exons in the output (see Figure 4.2.2).

ALTERNATE PROTOCOL

USING THE INTERACTIVE UNIX VERSION MZEF TO ANALYZE GENOMIC DNA SEQUENCES

Necessary Resources

Hardware

Any Unix or Linux workstation

Software

The appropriate MZEF interactive executable file (e.g., mzef)

The executable files for MZEF are free for academic users. The files may be downloaded from the cshl.org FTP site (see step 1 below). Commercial users and those who wish to obtain source codes (written in FORTRAN 77), should contact the CSHL licensing office (Dr. Carol Dempster, 516-367-6885, dempster@cshl.org).

The software has evolved into many different versions to meet the demands from different users. Consequently, there are several executable files available from the FTP site. The file names indicate the differences between the various forms. The default platform is SUN (Solaris) unless indicated explicitly at the end of an executable file name. The `1mb` *means the maximum input sequence size is 1 Mb, otherwise the maximum is 200 Kb. The* `cmd` *means all of the parameters must be entered from the command-line, other files are interactive (i.e., the program will prompt users for each parameter one line at a time during execution). The* `static` *means it does not require a run-time FORTRAN library, the default requires* `libF77.so.x` *libraries. The* `new` *or any versions after that (1997 or later) will not require files and data being in the current directory to run. Other versions may also be compiled at a special request to mzhang@cshl.org.*

Files

A FASTA file (*APPENDIX 1B*) with no more than 80 characters per line that contains the DNA sequence in which one wishes to identify the exons

MZEF can only take the standard DNA/RNA character symbols (either in capital or lower-case letters), ambiguous IUPAC symbols (APPENDIX 1A) will be converted to the standard symbols by a random draw (e.g., "N" will be converted into "A", "C", "G", "T" with equal probability).

The example used in the following is a 19-kb human genomic DNA sequence containing the serum albumin (ALB) gene (File name: `m12523.fasta`*; GenBank accession no. M12523, gi:178343; Minghetti et al., 1986). The sequence may also be found on the Current Protocols in Bioinformatics Web site at http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm. This gene has an alternative last exon, the CDS annotation is as follows:*

```
CDS  join(1776..1854,2564..2621,4076..4208,6041..6252,
     6802..6934,7759..7856,9444..9573,10867..11081,
     12481..12613,13702..13799,14977..15115,15534..15757,
     16941..17073,18526..18555)

CDS  join(1776..1854,2564..2621,4076..4208,6041..6252,
     6802..6934,7759..7856,9444..9573,10867..11081,
     12481..12613,13702..13799,14977..15115,15534..15757,
     16941..17073,17688..17732)
```

that may be compared with the MZEF predictions below.

The FORTRAN program also requires the following data files, which are available from the FTP site (see steps 1 and 2 below):

```
as1.dat
as2.dat
br1.dat
br2.dat
ds1.dat
ds2.dat
h6ex1.dat
h6ex2.dat
h6exc1.dat
h6exc2.dat
h6exi1.dat
h6exi2.dat
h6exl1.dat
h6exl2.dat
h6exr1.dat
h6exr2.dat
qda.dat
```

and `test.dat` is just a short input DNA sequence for a test run.

NOTE: *The names of the data files for each organism are the same, but the contents of the files differ.*

1. Create a new directory to hold the MZEF files, and change to that directory.

```
mkdir ~/MZEF
cd ~/MZEF
```

A copy of the FASTA file for the DNA sequence of interest (e.g., `m12523.fasta`*; see Necessary Resources) must be copied into the* `MZEF` *directory. For information on navigating through a Unix environment see APPENDIX 1C.*

If you intend to download the program and its associated data files for more than one organism, the directories should be named in a way that the user can keep track of the files; e.g., `MZEF_HUMAN` *in the case of the human data set.*

```
Internal coding exons predicted by MZEF
 File_Name: m12523.fas  Sequence_length:  19002  G+C_content:  0.350
  Coordinates    P    Fr1   Fr2   Fr3  Orf   3ss   Cds   5ss
  1817 -  1854 0.992 0.528 0.460 0.656 111 0.561 0.557 0.689
  2564 -  2621 0.635 0.636 0.522 0.374 112 0.443 0.557 0.567
  4076 -  4208 0.993 0.405 0.322 0.567 221 0.536 0.505 0.587
  6041 -  6252 0.971 0.391 0.596 0.513 212 0.538 0.573 0.646
  6802 -  6934 0.932 0.363 0.553 0.522 211 0.547 0.518 0.545
  7759 -  7856 0.999 0.642 0.592 0.481 112 0.536 0.622 0.607
  9444 -  9573 0.999 0.636 0.486 0.463 122 0.574 0.581 0.553
 10867 - 11081 0.997 0.626 0.493 0.403 122 0.541 0.558 0.597
 12481 - 12613 0.999 0.453 0.617 0.544 212 0.576 0.588 0.548
 13341 - 13425 0.757 0.414 0.498 0.556 221 0.460 0.537 0.719
 13702 - 13799 1.000 0.587 0.482 0.399 122 0.548 0.532 0.719
 14977 - 15115 0.864 0.502 0.591 0.509 212 0.526 0.574 0.457
 15534 - 15757 0.678 0.501 0.463 0.594 221 0.462 0.570 0.562
 16941 - 17073 0.999 0.667 0.451 0.515 122 0.483 0.613 0.609
 17812 - 17874 0.514 0.489 0.434 0.544 221 0.539 0.540 0.493
```

Figure 4.2.3 Prediction results from the interactive Unix version of MZEF (prior probability = 0.04; overlap = 0).

2. Download and install the appropriate MZEF executable file and all of the required data files. All of the files are available by running an FTP session as follows:

```
%ftp cshl.org
Name: anonymous
Password: [your internet address]
ftp> cd pub/science/mzhanglab/mzef
ftp> get README
ftp> cd human
ftp> binary
ftp> get mzef mzef_new
ftp> mget *.*
...[answer "yes" to all the files - this will download
the required data files]
ftp> quit
```

   The instructions on how to install MZEF are in the README file, which also has a brief description about the program and parameters.

   The command: `get mzef mzef_new` *downloads the executable* `mzef` *and renames it* `mzef_new`.

3. Change the permissions on the executable by issuing the following command:

```
chmod +rwx mzef_new
```

4. Run the interactive version of MZEF locally on a Unix/Linux machine. The results are shown in Figure 4.2.3.

```
%mzef_new
ENTER NAME OF THE SEQUENCE FILE (in single quotes)
'm12523.fasta'
ENTER 1 FOR FORWARD, 2 FOR REVERSE
1
ENTER PRIOR PROBABILITY (suggesting .04)
.04
ENTER OVER LAPPING NUMBER (suggesting 0)
0
```

See Critical Parameters for further discussion of these parameters.

For this example, the new prior probability value (Prior = 0.04) was used instead of the Web default (0.02), therefore, one can see some additional exon predictions in the output (exon 2564-2821 was missed in Basic Protocols 1 and 2 because when Prior = 0.02 its P score < 0.5; see Figures 4.2.1, 4.2.2, and 4.2.3).

GUIDELINES FOR UNDERSTANDING RESULTS

The result output contains the following information: File_Name (maybe truncated if too long), Sequence_length (in basepairs), G+C_content (see Feature Variables Used in MZEF in this unit's Appendix) and a table of internal coding exons predicted. The nine columns in the table are:

Coordinates–the exon coordinates in the input DNA sequence (if Strand = 2, one should reverse-complement each output region to get the sense-strand segment);
P–the posterior probability (>0.5) for each exon; how likely is it an exon;
Fr1–first-fame preference score; how likely the 1st frame is coding;
Fr2–second-frame preference score; how likely the 2nd frame is coding;
Fr3–third-frame preference score; how likely the 3rd frame is coding;
Orf–open reading frames, e.g., 112 (or 110) means the first and the second frames are open;
3ss–the acceptor site score (3′ splice-site score);
Cds–the coding-potential score (exon coding potential);
5ss–the donor site score (5′ splice-site score).

In the Web example (see Basic Protocol 1; Figure 4.2.1), the predicted exon in region (4076..4208) has only one ORF in the third frame, which is consistent with Fr3 being relatively larger than both Fr1 and Fr2. For the same reason, the predicted exon (7759..7856) has two ORFs (in the first and the second because Orf = 112), but the ORF in the first frame is more likely to be the real one because Fr1 is larger than Fr2.

Although MZEF does not assemble the exons into a gene model, occasionally, one can resolve the frame ambiguity or eliminate the false-positive exon predictions by requiring frame compatibility between adjacent coding exons. In the Web example above (see Basic Protocol 1; Figure 4.2.1), the predicted exon (6802..6934) had two ORFs (i.e., Orf = 211) with Fr2 (0.553) Fr3 (0.522), but in order for it to be compatible with the adjacent coding exons, the second ORF would have to be used. For similar reasons, the predicted exon (13341..13425) may be a false-positive because its ORF is not compatible with others and its P score is relatively low compared to that of the adjacent ones. One must be careful when using frame-compatibility because it assumes the adjacent ones are correct and there is no missing (false-negative) one next to it. Sometimes, a true exon's frame is not compatible to the next predicted one because of alternative splicing (i.e., it may be compatible with another one further downstream).

COMMENTARY

Background Information

MZEF is based on Quadratic Discriminant Analysis (QDA). QDA assumes real exons and pseudoexons are distributed as two different normal distributions in the feature space; it uses training data to construct the optimal discriminant surface (non-linear) to separate them (Zhang, 1997).

For a more detailed discussion of the theory behind MZEF, please see this unit's Appendix.

Advantages

MZEF is simple and fast. It is easily portable and may be incorporated into other programs readily. It can find internal coding exons in a short DNA sequence fragment that may not contain the full gene (it only requires a 54-bp flanking intron sequence). It can also output exons with alternative splice sites by allowing overlaps. It can handle very short exons (>18 bp) and tends to give better accuracy on exon-level statistics.

Limitations

Since MZEF is only designed to identify one class (albeit the most important class) of exons, internal coding exons, one would need other tools for identifying the other eleven classes (Zhang, 1998c) of exons (see Suggestions for Further Analysis). MZEF does not produce a gene model; one has to assemble a gene model by hand. This may not be regarded as a limitation when one is facing alternative splicing that occurs in nearly 60% of human genes (IHGSC, 2001; Modrek and Lee, 2002). The user cannot adjust various threshold values other than the few input parameters and must run the reverse strand separately.

Other options for similar analysis

There are two related programs that extend MZEF to improve performance under specific conditions. One is called GSA2 (X.Q. Huang, unpub.), which has combined MZEF with the EST database search results. It may be accessed at the AAT (Analysis and Annotational Tool) Web site *http://genome.cs.mtu.edu/aat/aat.html*. If one uses the same sequence and parameters as the example of the interactive MZEF run (see Alternate Protocol), one will obtain the results shown in Figure 4.2.4 from the AAT server.

It can be seen that the two false-positive internal coding exons, (13341..13425) and (17812..17874) have been eliminated due to the lack of EST matches. There is a danger that a novel exon may also be eliminated.

Another related program is called MZEF-SPC (Thanaraj and Robinson, 2000), which is an integrated system for exon finding with SpliceProximalCheck as a front-end for MZEF. It may be accessed at the EBI Web site *http://industry.ebi.ac.uk/~thanaraj/MZEF-SPC.html*. If one uses the same sequence and parameters as the example of the command-line MZEF run (see Figure 4.2.2), one will obtain the results shown in Figure 4.2.5 from the MZEF-SPC server.

Since Overlap was set to 1 (the default Overlap = 10 in the MZEF-SPC server) among overlapping MZEF predicted exons, MZEF-SPC was able to pick out most of the exons correctly except the last one. On average, however, MZEF-SPC should pick out more true exons among overlapping ones than MZEF nonoverlapping predictions. When selecting true exons among possible ones, frame compatibility should also be considered.

Critical Parameters and Troubleshooting

As mentioned above, MZEF requires three input parameters (other than the sequence file itself):

Strand = 1 or 2. One should try both strands if the coding strand information is unknown.

```
                              Prediction Results

Sequence: >GI|178343|GB|M12523.1|HUMALBGC HUMAN SERUM ALBUMIN (ALB)
GENE, COMPLETE CDS
Length: 19002 bp          C+G Content: 35%

Type      End5      End3   Leng Fr   St/Ac   Do/Te   FrCod    Prob     Score

Intr      1817      1854     38  2   0.561   0.689   0.656   0.992     10.02
Intr      2564      2621     58  0   0.443   0.567   0.155   0.635      1.41
Intr      4076      4208    133  2   0.536   0.587   0.567   0.993     10.15
Intr      6041      6252    212  1   0.538   0.646   0.691   0.971      7.38
Intr      6802      6934    133  1   0.547   0.545   0.553   0.932      5.51
Intr      7759      7856     98  0   0.536   0.607   0.965   0.999     14.92
Intr      9444      9573    130  0   0.574   0.553   0.636   0.999     14.18
Intr     10867     11081    215  0   0.541   0.597   0.991   0.997     12.02
Intr     12481     12613    133  1   0.576   0.548   0.617   0.999     13.75
Intr     13702     13799     98  0   0.548   0.719   0.976   1.000     19.63
Intr     14977     15115    139  1   0.526   0.457   0.591   0.864      3.90
Intr     15534     15757    224  2   0.462   0.562   0.724   0.678      1.79
Intr     16941     17073    133  0   0.483   0.609   0.667   0.999     14.48

Reverse Strand

Notations:
Star, initial exon; Intr, internal exon; Term, terminal exon;
End5, 5' exon coordinate; End3, 3' exon coordinate;
Leng, exon length; Fr, frame number (0, 1, or 2);
St/Ac, start or acceptor site score; Do/Te, donor or stop site score;
FrCod, in -frame coding score; Prob, exon     probability; Score, exon
score.
Coding cDNA and protein for each coding region:
```

Figure 4.2.4 Prediction results from AAT. Note that all the predictions are on the forward strand, and that no exons are predicted on the reverse strand.

P0 or Prior probability. It reflects the a priori belief on the coding exon density in the genomic region. As one can see from the above examples, when P0 was changed from 0.02 to 0.04, MZEF predicted two more exons that include one true exon (2564..2621) and another false-positive exon (17812..17874; see Figures 4.2.1 and 4.2.3). So the effect of increasing P0 is to have more putative exons predicted. The default value is 0.02 for the Web version and is 0.04 for the local version.

Overlap allows predicted exons to overlap. The default is 0, namely, overlapping is not allowed. As shown in the command-line version example above, when Overlap = 1 is set, at most, one overlapping exon is allowed to output for each exon region (see Figure 4.2.2). This allows the user to choose an exon with an alternative splice site, especially when one is looking for an exon that has a compatible frame with other adjacent exons during gene model building. Normally, if G+C_content is low, the exon density may also be low.

Results

The predicted exon boundaries by MZEF are further characterized by Splice Proximal Check.

Seq	Exon	Acceptor Site	Donor Site
gi|178343|gb|M11817-1854		Possibly TRUE	Possibly TRUE
gi|178343|gb|M14076-4208		Possibly TRUE	Possibly TRUE
gi|178343|gb|M16041-6252		Possibly TRUE	Possibly TRUE
gi|178343|gb|M16072-6252		FALSE	Possibly TRUE
gi|178343|gb|M16802-6934		Possibly TRUE	Possibly TRUE
gi|178343|gb|M17759-7856		Possibly TRUE	Possibly TRUE
gi|178343|gb|M19444-9573		Possibly TRUE	Possibly TRUE
gi|178343|gb|M19449-9573		FALSE	Possibly TRUE
gi|178343|gb|M110867-11081		Possibly TRUE	Possibly TRUE
gi|178343|gb|M110914-11081		Possibly TRUE	Possibly TRUE
gi|178343|gb|M112481-12613		Possibly TRUE	Possibly TRUE
gi|178343|gb|M112505-12613		FALSE	Possibly TRUE
gi|178343|gb|M113341-13425		Possibly TRUE	Possibly TRUE
gi|178343|gb|M113357-13425		Possibly TRUE	Possibly TRUE
gi|178343|gb|M113702-13799		Possibly TRUE	Possibly TRUE
gi|178343|gb|M113730-13799		FALSE	Possibly TRUE
gi|178343|gb|M114977-15115		Possibly TRUE	Possibly TRUE
gi|178343|gb|M115534-15757		Possibly TRUE	Possibly TRUE
gi|178343|gb|M116941-17073		FALSE	Possibly TRUE
gi|178343|gb|M116969-17073		FALSE	Possibly TRUE

Figure 4.2.5 Prediction results from MZEF-SPC.

In addition to the three-user controllable parameters, there are also a few hard-coded MZEF parameters:

Minimum ORF size = 18 bp, because shorter exons are extremely rare;
Maximum ORF size = 999 bp, which was chosen according to the longest internal coding exon in the training set;
Minimum acceptor site score = 0.38;
Minimum donor site score = 0.26;
Minimum total splice site score (acceptor site score + donor site score) = 0.79.

The purpose of setting such thresholds is to reduce the amount of false-positives and to cut down CPU time, perhaps at a reasonable expense of a few false-negatives.

Finally, MZEF can only output exons that have a P-value >0.5.

Most often, the troubleshooting should start by checking if the input sequence file format is correct (FASTA format; *APPENDIX 1B*). One should always check the sequence length in the output report and see if it is correct. If it is not correct, it is most likely caused by extra blank spaces or >80 character per line in the sequence file. One should always test the program with a gene of known structure. If the number of predicted exons is too small, try to increase the P0 and vice versa.

Suggestions for Further Analysis

One should always run several gene-finding programs, such as GENSCAN, FGENES, GRAIL, and others. Extensive research has shown that an exon predicted with a high score from more than two programs is most likely real, even if there is no cDNA support,

because the exon may only be expressed under special conditions. Homology searches against known gene databases are also indispensable.

MZEF should also be run in conjunction with other programs that can predict different types of exons and/or different parts of the gene structure. Often, the results from these programs can reinforce each other. For example, one could run CorePromoter (Zhang, 1998b), CpG_Promoter (Ioshikhes and Zhang, 2000), FirstEF (a first exon finder; Davuluri et al., 2001), JTEF (a last exon finder; Tabaska et al., 2001), and Polyadq (a polyA site finder; Tabaska and Zhang, 1999). All these programs can be accessed from *http://www.cshl.org/mzhanglab/*. Examples of how one can combine some of these programs for gene-finding may be found in Zhang (2000).

Internet Resources

http://www.cshl.org/genefinder

MZEF Web server

http://www.cshl.org/mzhanglab

Papers and other related information for MZEF

ftp://cshl.org/pub/science/mzhanglab

FTP site for MZEF

Literature Cited

Bishop, C. M. 1996. Neural Networks for Pattern Recognition. Oxford, Clarendon Press.

Box, G.E.P. and Cox, D.R. 1964. An analysis of transformations. *J. R. Statist. Soc. B* 26:211-252.

Chen, T. and Zhang, M.Q. 1998. POMBE: A fission yeast gene-finding and exon-intron structure prediction system. *Yeast* 14:701-710.

Davuluri, R., Grosse, I., and Zhang, M.Q. 2001. Computational identification of promoters and first exons in the human genome. *Nature Genet*. 29:412-417.

Fisher, R.A. 1936. The use of multiple measurements in taxonomic problems. *Ann. Eugen*. 7:179-188.

Fukunaga, K. 1990. Introduction to Statistical Pattern Recognition 2nd Edition. Academic Press, San Diego.

International Human Genome Sequencing Consortium. 2001. Initial sequencing and analysis of the human genome. *Nature* 409:860-921.

Ioshikhes, I. and Zhang, M.Q. 2000. Large-scale human promoter mapping using CpG islands discrimination. *Nature Genet*. 26:61-63.

Minghetti, P.P., Ruffner, D.E., Kuang, W.J., Dennison, O.E., Hawkins, J.W., Beattie, W.G., and Dugaiczyk, A. 1986. Molecular structure of the human albumin gene is revealed by nucleotide sequence within q11-22 of chromosome 4. *J. Biol. Chem*. 261:6747-6757.

Modrek, B. and Lee, C.A. 2002. A genomic view of alternative splicing. *Nat. Genet.* 30:13-19.

Solovyev, V.V., Salamov, A.A., and Lawrence, C.B. 1994. Predicting internal exons by oligonucleotide composition and discriminant analysis of spliceable open reading frames. *Nucl. Acids Res*. 22:5156-5163.

Tabaska, J.E. and Zhang, M.Q. 1999. Detection of polyadenylation signals in human DNA sequences. *Gene* 231:77-86.

Tabaska, J.E., Davuluri, R., and Zhang, M.Q. 2001. A novel 3′-terminal exon recognition algorithm. *Bioinformatics* 17:602-607.

Thanaraj, T.A. and Robinson, A.J. 2000. Prediction of exact boundaries of exons. *Briefings in Bioinformatics* 1:34356.

Zhang, M.Q. 1997. Identification of protein coding regions in the human genome by quadratic discriminant analysis. *Proc. Natl. Acad. Sci. U.S.A*. 94:565-568.

Zhang, M.Q. 1998a. Identification of protein-coding regions in *Arabidopsis thaliana* genome based on quadratic discriminant analysis. *Plant Mol. Biol*. 37:803-806.

Zhang, M.Q. 1998b. Identification of human gene core-promoters in silico. *Genome Res*. 8:319-326.

Zhang, M.Q. 1998c. Statistical features of human exons and their flanking regions. *Hum. Mol. Genet*.7:919-932.

Zhang, M.Q. 2000. Discriminant analysis and its application in DNA sequence motif recognition. *Briefings in Bioinformatics* 1:331-342.

Key References

Zhang, 1997. See above.

This is the original MZEF paper.

Zhang, 1998c. See above.

This has human exon classification and feature statistics.

Zhang, 2000. See above.

This is a tutorial on discriminant analysis and has examples on how to combine MZEF with other programs.

Contributed by Michael Q. Zhang
Cold Spring Harbor Laboratory
Cold Spring Harbor, New York

APPENDIX

Discriminant Analysis and Bayes Error

MZEF is based on a classical discrimination method QDA (Quadratic Discriminant Analysis), which is a direct descendant of LDA (Linear Discriminant Analysis). Discriminant analysis belongs to general statistical pattern recognition methods and has been widely used in many fields for optimal classification (e.g., Fukunaga, 1990). Discriminant analysis is used to answer the following question: given N objects, how can one assign each object into K known classes with minimum error? For simplicity, the case of $K = 2$ is only considered, although the theory can be easily generalized to $K > 2$. In order to distinguish one class object from another, one needs two things: a set of feature variables $x = \{x_\alpha: \alpha = 1,...,p\}$ and a decision rule (i.e., classifier) C such that given the measured values x^i for the ith object, C would be able to map it into either class I (denoted by "+") or class II (denoted by "–", see Figure 4.2.6). In practice, choosing the set of feature variables that is most discriminative with respect to the two classes is the key to success. For example, sex hormone level is a much better discriminative feature variable than weight when classifying people as males and females. Although there are many systematic methods for selecting better feature variables, it is still more or less like a black art, which depends heavily on the master's insight to the nature of the subject. Once the set of feature

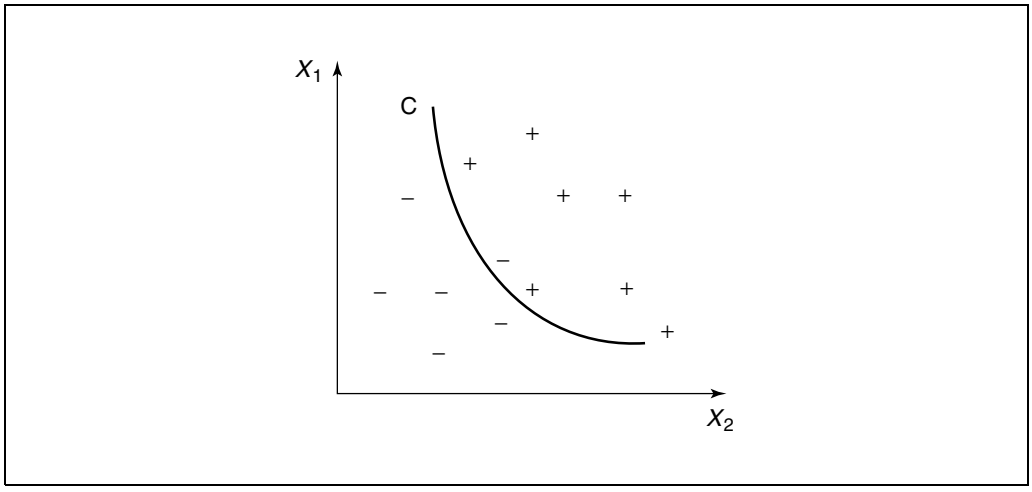


Figure 4.2.6 A classifier C separates $N = 13$ sample points in K = 2 feature space. Error = 1.

variables is decided (or given), one can represent the N objects to be classified as N sample points x^i in the p-dimensional feature space. Discriminant theory will offer the mathematical tools for finding the optimal classifier in the sense of minimizing the classification errors.

In general, the (Bayesian) theory assumes the sample points were drawn from two distinct distributions $p(x|+) = f_+(x)$ and $p(x|-) = f_-(x)$. If these conditional distributions and the a priori probabilities π_+ and π_- (for a randomly chosen sample being in class + or −, respectively) are known, then the a posteriori probability $q_+(x)$ of seeing the data x and it belonging to class + is given by the Bayes formula: $q_+(x) = \pi_+ f_+(x)/[\pi_+ f_+(x) + \pi_- f_-(x)]$, this is because $q_+(x) = p(+|x) = p(+,x)/p(x) = p(x|+)\pi_+/p(x) = p(x|+)\pi_+/[p(x|+)\pi_+ + p(x|-)\pi_-]$.

A discriminant function $h(x)$ is defined as the log likelihood ratio: $h(x) = \ln [q_+(x)/q_-(x)]$.

One can choose the decision boundary C_B (the *Bayes decision rule*) as the hyper-surface $h(x) = 0$, because for any given sample point x^i, it would be more likely to belonging to class + if $h(x^i) > 0$. By assigning x^i to class +, one would make an error with probability $q_-(x^i) < q_+(x^i)$. Similarly, by assigning x^j to class − when $h(x^j) < 0$, one would make an error with probability $q_+(x^j) < q_-(x^j)$. In general for any decision rule C, the total error (the Bayes error)

$$\varepsilon = \text{probability of misclassification} = \int_{R_+} q_-(x)dx + \int_{R_-} q_+(x)dx, \qquad \textbf{(4.2.1)}$$

where the regions R_+ and R_- are classified to + and − by C, respectively.

QDA and its Relation to LDA

When samples are assumed to be drawn from two different normal distributions

$$\begin{aligned} f_k(x) &= \frac{1}{(2\pi)^{p/2}|\Sigma_k|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1}(x-\mu_k)\right\} \\ &= \frac{1}{(2\pi)^{p/2}|\Sigma_k|^{1/2}} \exp\left\{-\frac{1}{2}\Delta^2(x-\mu_k)\right\} \end{aligned} \qquad \textbf{(4.2.2)}$$

where μ_k and Σ_k are the mean and the covariance matrix for the class k (k = + or −, $|\Sigma_k|$ is the determinant of the $p \times p$ matrix and $\Delta_k(x,y)$ is called Mahalanobis distance between two vectors x,y within class k, the discriminant function will be a quadratic function of x (through Δ^2 defined in Equation 4.2.2):

$$h(x) = -\frac{1}{2}\left[\Delta^2(x,\mu_+) - \Delta^2(x,\mu_-) + \ln\frac{|\Sigma_+|}{|\Sigma_-|}\right] + \gamma_\pm \qquad \textbf{(4.2.3)}$$

where $\gamma_\pm = \ln(\pi_+/\pi_-)$. Geometrically, the decision boundary is a quadratic hyper-surface in p-dimensions (Figure 4.2.7) when $\Sigma_+ \neq \Sigma_-$. Using such a quadratic discriminant function for classification is called QDA (quadratic discriminant analysis). When $\Sigma_+ = \Sigma_- = \Sigma$, the quadratic terms in $h(x)$ will be canceled out.

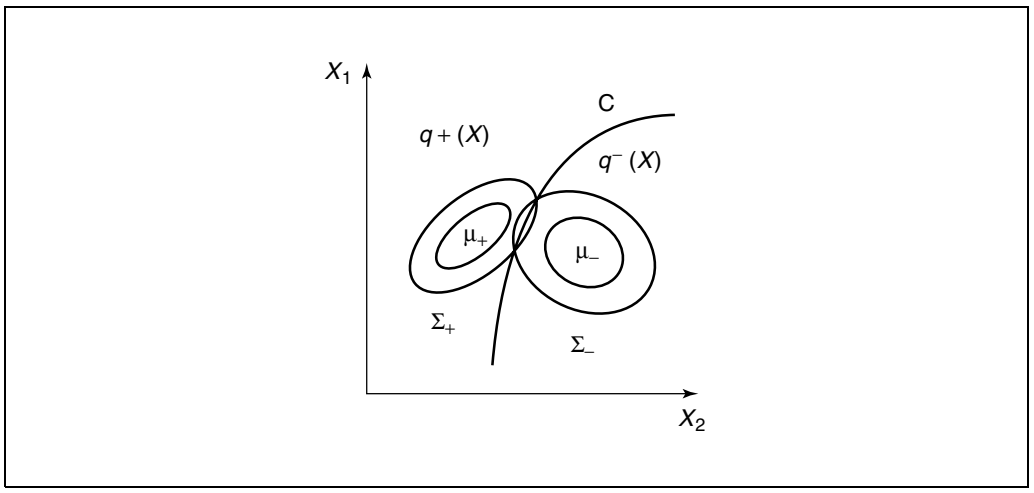


Figure 4.2.7 Quadratic decision boundary for normal distributions.

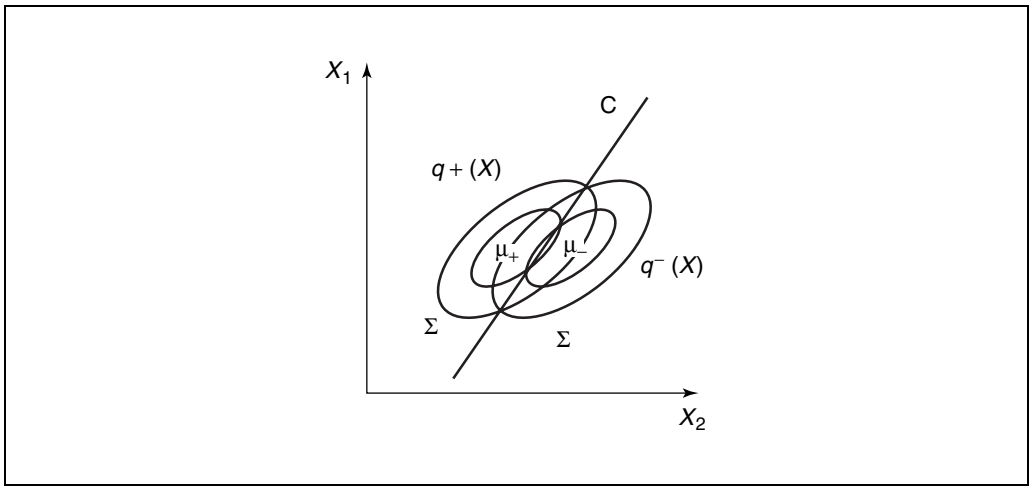


Figure 4.2.8 Linear decision boundary for normal distributions when $\Sigma_+ = \Sigma_-$.

$$h(x) = (\mu_+ - \mu_-)^T \Sigma^{-1} x - \frac{1}{2}(\mu_+^T \Sigma^{-1} \mu_+ - \mu_-^T \Sigma^{-1} \mu_-) + \gamma_\pm \qquad \textbf{(4.2.4)}$$

The Bayes decision boundary will become linear (hyper-plane as seen in Figure 4.2.8). Although linear decision boundaries are optimal (in the Bayes sense) only for normal distributions with equal covariance matrices, because of its simplicity, one may always want to know how well one can do with just a linear discriminant function for an arbitrary class of distributions. A general linear discriminant function can be written as $h(x) = V^T x + \nu$, which means x is projected onto a vector, V, and the variable $y = V^T x$ in the projected linear space is classified according to whether $y > \nu$ or $y < \nu$. Suppose the means and variances in the projected subspace are $\eta_\pm = E\{h(x)|\pm\} = V^T\mu_\pm + \nu$ and $\sigma_\pm^2 = \mathrm{Var}\{h(x)|\pm\} = V^T\Sigma_\pm V$, the most popular choice for the optimal V is the following Equation 4.2.5:

$$V = \left(\frac{1}{2}\Sigma_+ + \frac{1}{2}\Sigma_-\right)^{-1}(\mu_+ - \mu_-) \qquad \textbf{(4.2.5)}$$

which maximizes the Fisher criterion $(\eta_+ - \eta_-)^2/(\sigma_+ + \sigma_-)$ (Fisher, 1936). One notices that the Fisher coefficient (Equation 4.2.5) will reduce to that of (Equation 4.2.4) when $\Sigma_+ = \Sigma_-$, although minimization of the Fisher criterion cannot provide an optimal value for the constant threshold ν, which may be chosen by minimizing the classification errors in the linear subspace. Using a linear discriminant function (often the Fisher discriminant function) for classification is called LDA (linear discriminant analysis; see Solovyev et al., 1994).

In real applications, one normally does not know the distributions. One should always try to transform variables so that they are approximately normal (there are many techniques for doing this, for instance, the Box-Cox transformation, 1964). Even if one assumes some parametric distributions, estimation of the parameters using the training data is still necessary. LDA is more robust because it does not require normality of the distributions, and it has fewer parameters to be estimated. But if one has sufficient data and the decision boundary is intrinsically nonlinear (two class distributions have very different shapes as indicated by $\Sigma_+ \neq \Sigma_-$), QDA may be superior. Of course, there are also other nonparametric methods that are beyond the scope of this unit. Discriminant analysis can be done equally well by neural networks or machine learning approaches, where the decision boundary or the distribution parameters are estimated by iteration algorithms (Bishop, 1996); here the multivariate statistical approach for its analytical clarity is the focus.

Feature Variables Used in MZEF

If f_A is some frequency found in class A, the author defines a preference for A versus B (say, exons versus pseudoexons) to be the ratio $p_{A,B} = f_A/(f_A + f_B)$. It is clear that if $f_A << f_B$, the preference for A would be close to zero; if $f_A = f_B$, the preference for A would be $\frac{1}{2}$ (no preference). There are nine feature variables used in MZEF and they are computed for high or low (0.48 being the cutoff) G + C query sequences separately. Suppose f_exon and f_intron are frequencies for 6mers (or 3mers) in the exon and intron regions pre-computed from the training data, then these 9 feature variables computed on-the-flight are:

1. Exon_length score, $x_1 = \log_{10}(\text{bp})$;
2. Intron-exon_transition score, x_2 = average [(intron_preference to the left) − (exon_preference to the right)] = [(sum of $p_{\text{intron,exon}}$ over all overlapping 6mers in the 54-bp window to the left of 3′ss) − (sum of $p_{\text{exon,intron}}$ over all overlapping 6mers in the 54-bp window to the right of 3′ss)]/49;
3. Branch-site score, x_3 = maximum log likelihood branch-site score found in the window (-54,-3) relative to 3′ss using the pre-computed weight matrix;
4. 3′ss splice-site score, x_4 = position-dependent triplet preference for true_acceptor versus pseudo_acceptor in the window (-24,+3) using pre-computed 3mer weight matrices;
5. Exon_score, x_5 = average [6mer preference for exon versus intron] = (sum of $p_{\text{exon,intron}}$ over all overlapping 6mers in the exon window) / (exon length − 5);
6. Strand_score, x_6 = average [6mer exon preference for the forward strand versus the reverse] = [sum of f_exon(w)/(f_exon(w) + f_exon(w')) over all overlapping 6mers w in the exon window]/(exon length − 5), where the 6mer w' is the reverse complement of w;
7. Frame_score, $x_7 = \max_{i=0,1,2}$ (frame-specific 6mer preference for exon versus intron in frame i in the exon window);

8. 5′ss splice-site score, x_8 = position-dependent triplet preference for true_doner versus pseudo_doner in the window (-3,+8) using pre-computed 3mer weight matrices;

9. Exon-intron_transition, x_9 = average [(exon_preference to the left) − (intron_preference to the right)] = [(sum of $p_{\text{exon,intron}}$ over all overlapping 6mers in the 54-bp window to the left of 5′ss) − (sum of $p_{\text{intron,exon}}$ over all overlapping 6mers in the 54-bp window to the right of 5′ss)]/49.

For the Arabidopsis_MZEF (Zhang, 1998a), a 60-bp flanking intron window is used (instead of 54-bp). Since no-isochore is found in *Arabidopsis* genome, no G+C specific feature variables are necessary. Because of the G+C content feature itself had been recognized as the important variable, Arabidopsis_MZEF introduced one additional feature variable − GC_ratio score, x_{10} = (G+C content in the exon)/(G+C content in the flanking introns).

Using geneid to Identify Genes

The gene-prediction program geneid is based on a simple hierarchical design: (1) search splicing signals, start codons, and stop codons, (2) build and score candidate exons, and (3) assemble genes from the exons (Guigó et al., 1992; Parra et al., 2000). Geneid was one of the first computational gene identification programs: an early version of geneid was available as an e-mail server in 1991 (Guigó et al., 1992). A new implementation of the system was written and released in 1999 (geneid v1.0; Parra et al., 2000). In 2002, new capabilities were added to geneid in order to include external information that supports genomic reannotation procedures (geneid v1.1). A new, more powerful version of geneid (geneid v1.2) was released in 2004, including new parameter configurations for a larger number of species. This version, while having an accuracy comparable to the most accurate gene-prediction programs, is very efficient at handling large genomic sequences in terms of memory and speed. geneid is able to analyze chromosome-size sequences in a few minutes on a standard workstation, and has a rich set of output options, which allow for a detailed analysis of gene features in genomic sequences. Both a Web server interface and a stand-alone distribution are available.

This unit describes how to use the geneid Unix application to predict genes along genomic sequences (see Basic Protocol 1). These can be multiple genes on both strands of large genome sequences, or partial genes or exon signals in small genomic fragments. Basic Protocol 1 describes the default behavior of geneid, and introduces the basic options for configuring its output. Next, options for visualizing the output are described (see Basic Protocol 2). A third protocol describes how to use geneid together with experimental evidence (or evidence coming from other sources) to reannotate sequences whose genomic features have been partially annotated (see Basic Protocol 3). Use of the Web server version of geneid is described in the Alternate Protocol. The Support Protocol describes how to download the geneid software, which is in the public domain under a GNU-GPL license (*http://www.gnu.org/*). Complete, up-to-date documentation is provided with the geneid distribution, and can also be accessed through the geneid Web page (see Support Protocol).

USING THE geneid UNIX APPLICATION TO PREDICT GENES

BASIC PROTOCOL 1

geneid can be used in two different ways: via a Web server (see Alternate Protocol), or as a Unix application. The best way to take full advantage of the different options available in geneid is by running the stand-alone program on a Unix workstation.

In both cases, the user provides an input DNA sequence as a FASTA file (*APPENDIX 1B*), and selects a suitable model of parameters depending on the species (or taxonomic group) from which the sequence originates. A number of options are available to configure geneid actions and output. Although this option is not directly available in the stand-alone Unix application, geneid output can be directly plugged into a number of publicly available visualization tools (see Basic Protocol 2).

This protocol describes the use of geneid as a stand-alone Unix application. For use of the geneid Web server as an alternative, see Alternate Protocol.

Necessary Resources

Hardware

Unix/Linux workstation with at least 256 Mb RAM (recommended)

Contributed by Enrique Blanco, Genís Parra, and Roderic Guigó
Current Protocols in Bioinformatics (2007) 4.3.1-4.3.28
Copyright © 2007 by John Wiley & Sons, Inc.

Software

geneid v1.2 full distribution (see Support Protocol)

Files

This protocol uses the following file: `example1.fa` (32 kb, masked)
It is a human genomic sequence extracted from the UCSC human genome browser (assembly March 2006; location: human chromosome 21, coordinates 13,903,812-13,935,812). This sequence can also be found at the samples subdirectory within the geneid distribution (see Support Protocol) and on the *Current Protocols in Bioinformatics* Web site at *http://www.currentprotocols.com*.

NOTE: In a Unix system, the syntax to use geneid is:

```
geneid [options] -P parameter_file input_sequence
```

where `parameter_file` is a file containing gene-model parameters for a given species (or taxonomic group), which the user normally downloads with the geneid distribution, and `input_sequence` is a file containing a DNA sequence in FASTA format (*APPENDIX 1B*). A number of options allow modification of the geneid default behavior.

The following assumes that geneid has been successfully installed in a directory within the file system (the `geneid` directory), and that this directory is the current working directory (see Support Protocol).

NOTE: An introduction to the Unix environment can be found in *APPENDIX 1C*.

1. Run geneid on the first example (`example1.fa`) with default options:

```
%geneid -P param/human3iso.param samples/example1.fa
```

geneid is a Unix command-line program that requires as input a file containing a DNA sequence in FASTA format (`samples/example1.fa`*; see APPENDIX 1B for discussion of FASTA format), and a parameter file. This is specified by using the option* `-P` *followed by the name of the parameter file. geneid provides parameter files for human (this example), Drosophila melanogaster, and other species in the directory* `param/`*. By default, geneid produces results in plain text, which are sent to the standard output (Unix terminal). These can then be redirected to a file or another program. In particular, they can serve as input to programs producing graphical visualization of genomic annotations (e.g., gff2ps or apollo, UNIT 9.5) or genome browsers that display such information on the Web (e.g., UCSC genome browser, UNIT 1.4; ENSEMBL, UNIT 1.15, etc.). The interaction between geneid and these systems is shown in Basic Protocol 2.*

2. Examine the results returned by geneid.

By default, geneid output consists of a series of genes predicted along the input sequence. geneid uses its own default output format. Other, more standard, formats can be specified via command-line options (see steps 5, 6, and 7). Predicted genes are described as lists of potential coding exons. For sequence example1, geneid predicts an eight-exon gene (see Fig. 4.3.1 for plain text output and Figs. 4.3.5, 4.3.6, and 4.3.7 for graphical representations).

Each exon is defined by a start signal (start codon or acceptor site), an end signal (donor site or stop codon), the strand, and the frame. Each exon (as well as each signal) is assigned a score. The score depends on the scores of the defining sites, and on the nucleotide composition of the exon sequence, measuring the likelihood of the exon (see Background Information). The score of a gene is the sum of the scores of its exons. geneid predicts only the coding fraction of a gene. Thus, geneid defines four classes of exons—First, Internal, Terminal, and Single (corresponding to single-exon or intronless genes). A multiexon gene starts with First exon (start codon to donor site), followed by any number

```
## date Wed Jan 17 18:01:10 2007
## source-version: geneid v 1.2 -- geneid@imim.es
# Sequence example1 - Length = 32001 bps
# Optimal Gene Structure. 1 genes. Score = 16.70
# Gene 1 (Forward). 8 exons. 470 aa. Score = 16.70
   First      736     1130       6.14   + 0 2    4.86    1.91   17.70    0.00   AA   1:132 example1_1
Internal     5504     5618       0.49   + 1 0    2.72    3.11    6.23    0.00   AA 132:170 example1_1
Internal     5778     5951       1.13   + 0 0    1.43    0.86   13.14    0.00   AA 171:228 example1_1
Internal     8730     8836       0.84   + 0 2    4.75    3.34    3.72    0.00   AA 229:264 example1_1
Internal    13186    13256       0.46   + 1 1    1.21    5.38    5.00    0.00   AA 264:288 example1_1
Internal    21287    21488       2.78   + 2 2   -2.27    5.44   15.95    0.00   AA 288:355 example1_1
Internal    29896    30019       1.56   + 1 0    1.27    0.56   14.90    0.00   AA 355:396 example1_1
Terminal    31726    31947       3.30   + 0 0    1.78    0.00   19.32    0.00   AA 397:470 example1_1

>example1_1|geneid_v1.2_predicted_protein_1|470_AA
MGTSGDHDDSFMKMLRSKMGKCCRHCFPCCRGSGTSNVGTSGDHENSFMKMLRSKMGKWC
CHCFPCCRGSGKSNVGAWGDYDHSAFMEPRYHIRREDLDKLHRAAWWGKVPRKDLIVMLR
DTDMNKRDKEKRTALHLASANGNSEVVQLLLDRRCQLNVLDNKKRTALIKAIQCQEDECV
LMLLEHGADRNIPDEYGNTALHYAIYNEDKLMAKALLLYGADIESKNKCGLTPLLLGVHE
QKQQVVKFLIKKKANLNVLDRYGRICELLSDYKEKQMLKISSENSNPVITILNIKLPLKV
EEEIKKHGSNPVGLPENLTNGASAGNGDDGLIPQRRSRKPENQQFPDTENEEYHSDEQND
TRKQLSEEQNTGISQDEILTNKQKQIEVAEQKMNSELSLSHKKEEDLLRENSVLQEEIAM
LRLELDETKHQNQLRENKILEEIESVKEKTDKLLRAMQLNEEALTKTNI*
```

Figure 4.3.1 Default geneid prediction on sequence example1. The fields, from left to right, are defined in Table 4.3.1.

(possibly zero) of Internal exons (acceptor site to donor site), and ends with a Terminal exon (acceptor site to stop codon). An intronless gene is constituted by a Single exon (start codon to stop codon).

Lines starting with the # character do not correspond to coding exons, but provide additional information about the prediction. At the top of the output, two lines starting with the characters ## display general information on the geneid process. After this main header, the line beginning with `# Sequence` *displays the name and the length of the input sequence whereas the line starting with* `#Optimal Gene Structure` *contains the number of genes predicted along the input sequence as well as the total score of the prediction, which is the sum of the scores of the predicted genes. Then, lines starting with* `# Gene` *provide general information on each gene: gene identifier, strand (forward or reverse), number of exons, gene product length, and gene score. After this, there is a line for each coding exon in the gene with the fields (from left to right) defined as in Table 4.3.1.*

After the set of lines corresponding to the exons of the predicted gene, the amino acid sequence of the gene is printed in FASTA format (APPENDIX 1B).

The frame and remainder (see Table 4.3.1) of an exon are the number of hanging nucleotides not included in complete codons at the left/right ends of exons when these are assembled into a gene. The formal definition of geneid frame is "The number of nucleotides (0,1,2) from the first nucleotide in the exon to the first nucleotide in the first complete codon in the same exon." The remainder is defined in geneid as "The number of nucleotides left (0,1,2) after the last complete codon has been translated from the exon sequence, given its frame." By definition, then, all First exons have frame 0 (as in Fig. 4.3.2), and all Terminal exons have remainder 0.

3. Obtain the set of predicted Start codons along the input sequence by typing:

```
%geneid -P param/human3iso.param -bo samples/example1.fa
```

In addition to the predicted genes, geneid provides a number of options which allow the investigator to print an exhaustive list of all the sequence signals and exons predicted along the query sequence (most of which are not included in the final gene prediction). This option can be useful, for instance, to carry out a detailed analysis of a small genomic region for potential alternative splice sites. If only information on these sites and exons is required, it may be advisable to use the option `-o`*, which switches off the gene-assembly engine, therefore consuming a smaller amount of memory and running time.*

Table 4.3.1 Information Provided by geneid for Each Coding Exon in the Gene

Fields[a]	Description
(1)	Type of exon (First, Internal, Terminal or Single)
(2,3)	Location of the exon within the input sequence (i.e., the positions of the exon-defining signals)
(4)	Score of the exon
(5)	Strand (+/–) of the gene (always the same for all the exons in the same gene)
(6)	Frame
(7)	Remainder
(8,9)	Score of the two signals defining the exon (start/acceptor and donor/stop)
(10)	Score derived from the nucleotide composition of the exon sequence
(11)	Score derived from potential similarity of the exon sequence to known coding sequences
(12-13)	Location of the exon within the amino acid sequence of the predicted gene
(14)	Gene identifier

[a]From left to right in Figs. 4.3.1 and 4.3.2 (bottom).

```
## date Wed Jan 17 18:13:51 2007
## source-version: geneid v 1.2 -- geneid@imim.es
# Sequence example1 - Length = 32001 bps
# Starts(+) predicted in sequence example1: [0,32000]
   Start       53       55      -2.05   +       CTGCACCACGTGCAATGNNN
   Start      156      158       0.59   +       NNNNNNNNNNNNNNATGACG
   Start      178      180      -2.09   +       TTGGCTTACCTGTAATGGGT
   Start      610      612      -1.80   +       GCTGTTAAAAGCAGATGGTG
   Start      631      633      -2.48   +       CTGAGGTTTGTTCAATGCCC
   Start      679      681       2.60   +       ATCTCAGGAGCAAGATGGGC

## date Wed Jan 17 18:15:02 2007
## source-version: geneid v 1.2 -- geneid@imim.es
# Sequence example1 - Length = 32001 bps
# Firsts(+) predicted in sequence example1: [0,32000]
   First      156      174      -6.10   + 0 1    0.59   -1.40   -1.54    0.00   7       MTAWLTt
   First      156      181      -5.74   + 0 2    0.59   -0.54   -1.94    0.00   9       MTAWLTCNgg
   First      156      197     -11.55   + 0 0    0.59   -6.71   -7.21    0.00   14      MTAWLTCNGCASLC
   First      178      181      -6.58   + 0 1   -2.09   -0.54    0.00    0.00   2       Mg
   First      178      197     -10.49   + 0 2   -2.09   -6.71   -0.53    0.00   7       MGALRFgc
   First      178      216     -11.93   + 0 0   -2.09   -6.71   -4.12    0.00   13      MGALRFAFLAWPC
```

Figure 4.3.2 Predicted Start codons (top) and First exons (bottom) on sequence example1 (partial output). The fields, from left to right, are defined in Table 4.3.1 and steps 3 and 4 of Basic Protocol 1.

In the example, the results of which are shown in Figure 4.3.2 (top), potential start codons are displayed by using the option `-b`. *Other signals such as Stop codons, Acceptor splice sites, or Donor splice sites can be printed using the options* `-e`, `-a`, *or* `-d`, *respectively. All options can be specified at once—e.g.,* `geneid --bead` *(in any order)—and geneid then produces the exhaustive list of all potential sequence signals. For large (and not so large) genomic sequences, this can produce very large outputs.*

Each signal is printed in a separate record (line) with the following fields: type of signal, position, score, strand, and signal sequence. As geneid internally splits the input sequence into consecutive fragments, signals found both in forward and reverse strands are displayed for every fragment, after a header specifying the fragment positions in the input sequence.

4. Obtain the set of predicted First exons along the input sequence by typing:

```
%geneid -P param/human3iso.param -fo samples/example1.fa
```

geneid can also print all candidate exons along the query sequence. The options `-f`, `-i`, `-t`, *and* `-s` *are provided to print the predicted exons of each class (First, Internal, Terminal, and Single). The options can be combined to print more than one class of exons. In such a case, exons are printed separately by class. If exons of all classes are to be printed, it is advisable to use just the option* `-x` *which prints the list of all exons sorted by position. As shown in Figure 4.3.2 (bottom), each predicted exon is printed in a separate record containing the fields 1 to 11 as described in Table 4.3.1, plus the length and the amino acid sequence of the exon.*

5. Obtain a more complete output by using the option `-X`:

```
%geneid -P param/human3iso.param -X samples/example1.fa
```

By using the option `-X`, *geneid produces a more exhaustive output of the gene prediction. Each exon is now described in three different lines (Fig. 4.3.3). The first one describes the exon start signal (as in step 3), the second line describes the exon itself (as in step 4), and the third line describes the exon end signal (as in step 3).*

6. Obtain the predicted genes in the GFF format by using the option `-G`:

```
%geneid -P param/human3iso.param -G samples/example1.fa
> geneid_output.gff
```

General Feature Format or GFF (http://www.sanger.ac.uk/Software/formats/GFF/) is a proposed standard format for describing genes and other features associated with DNA, RNA, and protein sequences. Each feature is described as a list of fixed fields or columns delimited by tabs. This format is very easy to parse by bioinformatics applications. There are a number of tools, including visualization ones, that can process GFF files (see Basic Protocol 2). geneid produces GFF-compliant output with the option `-G`. *This option can be applied to any set of gene features selected to be printed (as in steps 2 and 3). The result is shown in Figure 4.3.4. A set of standardized* `##` *lines appear at the top of the GFF*

```
## date Wed Jan 17 18:28:20 2007
## source-version: geneid v 1.2 -- geneid@imim.es
# Sequence example1 - Length = 32001 bps
# Optimal Gene Structure. 1 genes. Score = 16.70
# Gene 1 (Forward). 8 exons. 470 aa. Score = 16.70
   Start      736     738       4.86   +       GCGGCAAGAGCAACATGGGC
   First      736    1130       6.14   + 0 2    4.86    1.91   17.70    0.00   AA   1:132 example1_1
   Donor     1130    1131       1.91   +       GAGGTAACC
Acceptor     5503    5504       2.72    0.00    0.00   +       CCTCAAGTCTTCTCACTCTCATAGGAC
Internal     5504    5618       0.49   + 1 0    2.72    3.11    6.23    0.00   AA 132:170 example1_1
   Donor     5618    5619       3.11   +       AAGGTATGC
Acceptor     5777    5778       1.43    0.00    0.00   +       TTGTTTTTGGTCTAATACTGACAGGCC
Internal     5778    5951       1.13   + 0 0    1.43    0.86   13.14    0.00   AA 171:228 example1_1
   Donor     5951    5952       0.86   +       AAGGTATAG
Acceptor     8729    8730       4.75    0.00    0.00   +       TTTTTGTTTTATATTGTTTTACAGTGT
Internal     8730    8836       0.84   + 0 2    4.75    3.34    3.72    0.00   AA 229:264 example1_1
   Donor     8836    8837       3.34   +       AAGGTATGG
Acceptor    13185   13186       1.21    0.00    0.00   +       GGTTTATTACATTTTTATACATAGAAT
Internal    13186   13256       0.46   + 1 1    1.21    5.38    5.00    0.00   AA 264:288 example1_1
   Donor    13256   13257       5.38   +       CAGGTAAGA
Acceptor    21286   21287      -2.27    0.00    0.00   +       AAAGGTTATTTTTATTCAATAAAGTGA
Internal    21287   21488       2.78   + 2 2   -2.27    5.44   15.95    0.00   AA 288:355 example1_1
   Donor    21488   21489       5.44   +       CAGGTAAGC
Acceptor    29895   29896       1.27    0.00    0.00   +       CATGAAAATTAAATTTTCTTCTAGTGA
Internal    29896   30019       1.56   + 1 0    1.27    0.56   14.90    0.00   AA 355:396 example1_1
   Donor    30019   30020       0.56   +       GAGGTATTT
Acceptor    31725   31726       1.78    0.00    0.00   +       TAAAAGTTCCCTTTGTTTACTTAGCTT
Terminal    31726   31947       3.30   + 0 0    1.78    0.00   19.32    0.00   AA 397:470 example1_1
    Stop    31945   31947       0.00   +       TTAA

>example1_1|geneid_v1.2_predicted_protein_1|470_AA
MGTSGDHDDSFMKMLRSKMGKCCRHCFPCCRGSGTSNVGTSGDHENSFMKMLRSKMGKWC
CHCFPCCRGSGKSNVGAWGDYDHSAFMEPRYHIRREDLDKLHRAAWWGKVPRKDLIVMLR
DTDMNKRDKEKRTALHLASANGNSEVVQLLLDRRCQLNVLDNKKRTALIKAIQCQEDECV
LMLLEHGADRNIPDEYGNTALHYAIYNEDKLMAKALLLYGADIESKNKCGLTPLLLGVHE
QKQQVVKFLIKKKANLNVLDRYGRICELLSDYKEKQMLKISSENSNPVITILNIKLPLKV
EEEIKKHGSNPVGLPENLTNGASAGNGDDGLIPQRRSRKPENQQFPDTENEEYHSDEQND
TRKQLSEEQNTGISQDEILTNKQKQIEVAEQKMNSELSLSHKKEEDLLRENSVLQEEIAM
LRLELDETKHQNQLRENKILEEIESVKEKTDKLLRAMQLNEEALTKTNI*
```

Figure 4.3.3 geneid prediction in extended format.

```
## gff-version 2
## date Wed Jan 17 18:33:02 2007
## source-version: geneid v 1.2 -- geneid@imim.es
# Sequence example1 - Length = 32001 bps
# Optimal Gene Structure. 1 genes. Score = 16.70
# Gene 1 (Forward). 8 exons. 470 aa. Score = 16.70
example1        geneid_v1.2     First            736    1130    6.14   +       0       example1_1
example1        geneid_v1.2     Internal        5504    5618    0.49   +       1       example1_1
example1        geneid_v1.2     Internal        5778    5951    1.13   +       0       example1_1
example1        geneid_v1.2     Internal        8730    8836    0.84   +       0       example1_1
example1        geneid_v1.2     Internal       13186   13256    0.46   +       1       example1_1
example1        geneid_v1.2     Internal       21287   21488    2.78   +       2       example1_1
example1        geneid_v1.2     Internal       29896   30019    1.56   +       1       example1_1
example1        geneid_v1.2     Terminal       31726   31947    3.30   +       0       example1_1
```

Figure 4.3.4 geneid prediction in GFF format.

file (the GFF header). Then, following the same structure as geneid default format, lines starting with the character # (assumed to be free-format comments in GFF), are used to provide general information on each gene predicted. GFF records provide information about the predicted gene features (from left to right): sequence name, source (the gene prediction program geneid in this case), feature (type of exon), start and end positions, score, strand, frame, and group (gene to which the exon belongs).

7. Obtain the same output in the XML format using the command:

```
%geneid -P param/human3iso.param -M samples/example1.fa
```

Extensible Markup Language or XML (http://www.w3.org/XML/) is a language developed from the experience obtained in the creation of SGML (Standard Generalized Markup Language) and HTML (Hypertext Markup Language), which is more widely used on the Internet. XML is basically a format to transfer information between computer programs (non-human-readable). Many parsing and displaying methods are available for XML, which makes it a powerful format to create Web documents. geneid supports XML format for predicted genes by means of the option `-M`*. The DTD (Document Type Definition) of geneid XML documents can be printed with the option* `-m`*.*

8. Examine the complete list of available options by using the option `-h`:

```
%geneid -h
```

The most relevant options that have not been discussed are:

`-v` *(verbose): This produces real-time detailed information while geneid is processing the input sequence.*

`-W -C` *(forward / reverse): This forces prediction in only one strand of the sequence.*

`-D` *(CDS sequence): This prints the DNA coding sequence of predicted genes.*

`-O -R -S` *(external features): By means of these options, additional information can be provided to geneid in order to modify the "ab initio" prediction (see Basic Protocol 3).*

BASIC PROTOCOL 2

VISUALIZING geneid PREDICTIONS

geneid does not produce a graphical output by itself. However, because it is capable of producing GFF output, several GFF visualization tools can be very easily used to display geneid predictions. This protocol describes how to use three different tools, the programs gff2ps and apollo (*UNIT 9.5*), and the UCSC genome browser (*UNIT 1.4*), to visualize the output produced by geneid. gff2ps (Abril and Guigó, 2000) was developed at the Institut Municipal d'Investigacio Medica (IMIM) in Barcelona as a visualization tool for genomic sequence annotations. gff2ps takes the annotated features on a genomic sequence in GFF format as input, and produces a high-quality PostScript file. It can be used in a very simple way, because gff2ps assumes that the GFF file itself carries enough formatting information, although it also allows, through a number of options and

a configuration file, a great degree of customization. apollo (*UNIT 9.5*; Lewis et al., 2002) is a genomic annotation viewer and editor. It has been developed as a collaboration between the Berkeley Drosophila Genome Project and The Sanger Centre (Cambridge, U.K.). apollo has been designed to be a complete genome annotation tool for use as a graphical front end to a database that stores sequence annotations. Its interactive interface allows the user to browse among predictions in a very intuitive way. Zoomable and scrollable display options are optimized to deal with large genomic regions. The UCSC genome browser (Hinrichs et al., 2006) is a Web tool developed at the University of California (Santa Cruz) to display on the Internet the available annotations on multiple genomes such as human, mouse, or fly. The UCSC genome browser shows the information in different tracks, each one containing different types of genomic information (e.g., genes, CpG islands, ESTs, etc.). Apart from the annotations provided by the system, the users can easily import predictions in GFF format. Thus, user data and genomic annotations are displayed in the same range of chromosomic coordinates. ENSEMBL (*UNIT 1.15*) and other genome browsers can also be used in a similar way to display geneid predictions.

Necessary Resources

Hardware

gff2ps: Unix/Linux workstation
apollo: Unix/Windows/Macintosh workstation with at least 164 Mb RAM (recommended)
UCSC browser: A computer and a connection to the Internet

Software

gff2ps (see Support Protocol for obtaining gff2ps)
apollo (see *UNIT 8.5* and Support Protocol for obtaining apollo)
UCSC Browser: Unix/Windows text editor (see *APPENDIX 1C*), an Internet browser: e.g., Firefox or Internet explorer

Files

geneid output (see Basic Protocol 1, step 6)

Visualization using gff2ps

1a. Run gff2ps on the geneid output extracted from Basic Protocol 1, step 6, to obtain a high-quality graphical output using the command:

```
%gff2ps geneid_output.gff > geneid_output.ps
```

gff2ps is a Unix command-line program which reads a GFF file. It produces PostScript output, which can be redirected to a file. The contents of the file can be displayed by means of programs such as ghostview or xpsview, or they can be directly sent to a PostScript printing device.

Figure 4.3.5 shows the default gff2ps output for the prediction obtained in Basic Protocol 1, step 6. The plot is fitted into a single block (assuming the length of the sequence to be the end of the most downstream feature), which is printed so as to fit into a single physical page. Genes predicted on the forward strand are displayed above the central bar and genes predicted in the reverse strand are displayed below. Exons are plotted with a height proportional to their score, using a three-color code schema. The color of the upstream half in the exon denotes the exon frame, and the color of the downstream half, the remainder. Nonoverlapping exons are frame-compatible if the remainder of the upstream exon matches the frame of the downstream one. gff2ps output can be highly customized. Users are therefore encouraged to develop their own configuration files to suit their specific needs. In particular, gff2ps can also be used to plot exhaustive predictions of potential sites and exons along the query sequence. In such a case, users are advised to process the geneid GFF output file and use the feature field as the source (see gff2ps user manual for details).

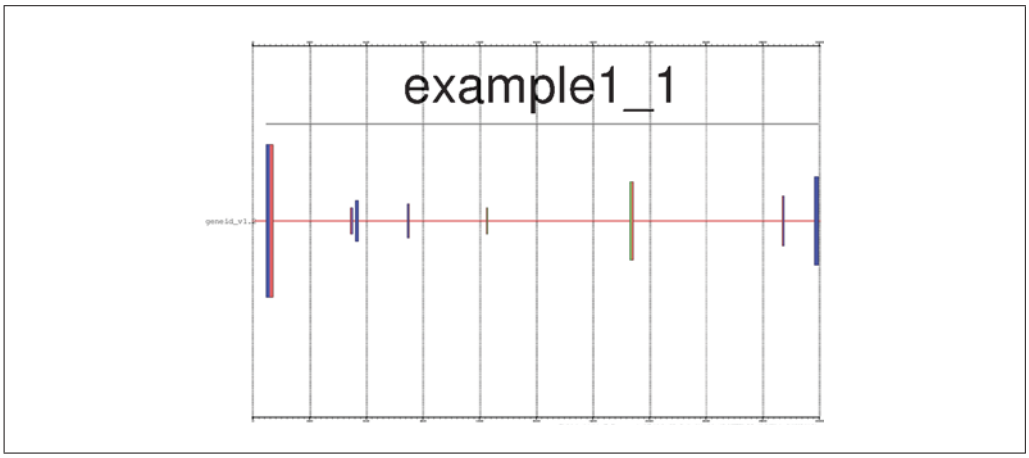


Figure 4.3.5 Using gff2ps to visualize geneid output. Graphical representation of geneid output on sequence example1 with default gff2ps.

Visualization using apollo

1b. To start an apollo session type:

```
%Apollo
```

The "load data" window will appear.

2b. Choose Ensembl GFF file format as a data source. Select a file for the visualization by writing the entire path in the GFF file box or by browsing the directory tree. Select `geneid_output.gff`, then click the OK button.

The main apollo window will appear (Fig. 4.3.6 shows the default apollo display of the prediction obtained in Basic Protocol 1, step 6). Coding exons provided by geneid are displayed below the main toolbar. Exons predicted on the forward strand are displayed

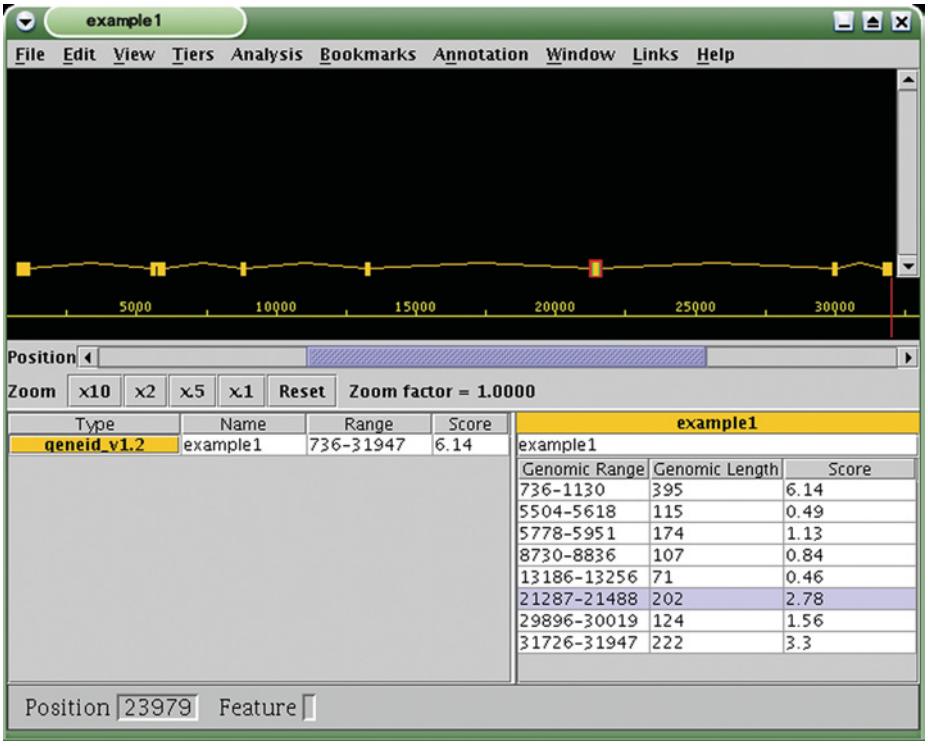


Figure 4.3.6 Using Apollo to visualize geneid output.

above the central scale bar, showing the nucleotide coordinates, and exons predicted in the reverse strand are displayed below. Exons belonging to the same gene are joined together by a line. Zoom in using the ×10 and ×2 buttons and zoom out using the ×.1 and ×.5 buttons. Use the scroll bar to move along the sequence. The detail panel underneath the main panel shows information about any feature or set of features selected. The left-hand panel shows the type (and color) of the feature, the name, the range, and the score. The right-hand panel shows more information about each individual exon (genomic range, genomic length, and score). Selecting an individual exon in the main window causes the exon to be selected in the right-hand panel for an easier identification, and vice versa.

Visualization using the UCSC genome browser

1c. In order to import the geneid GFF output in the UCSC genome browser, it is necessary to first locate the correct chromosomic location in the human genome for the sequence `example1.fa`. Load the UCSC genome browser Web page and click over the Genome Browser link. Select the human genome. Then, introduce the coordinates of this region in the Position box: chr21:13,903,812-13,935,812 and press the Submit button.

The main UCSC genome window will appear. The basic unit of information in the UCSC genome browser is the track. Each track is a graphical line of annotations in the image. Graphical features are associated to text information that contains the coordinates of the different genomic elements. Boxes and other elements in the image contain additional information that is shown according to the level of detail selected in the options below the picture. Zoom in and out using the 10×, 3×, and 1.5× buttons. Use the scroll buttons to move along the sequence.

2c. The UCSC genomic features are displayed along the chromosomes. Then, the geneid GFF output must be adapted to fit the range of the current window. The translation of the geneid predictions into the UCSC genomic window can be easily obtained by adding the value 13,903,812 to each coordinate in the geneid output. This conversion can be performed with a text editor.

On Unix systems, this operation can be easily performed using any standard file-editing tool (e.g., `awk` *or* `sed`*). For example, the* `awk` *command would be:*

```
%awk'BEGIN{OFS="\t"}{print $1,$2,$3,$4+13903812,$5+13
903812, $6,$7,$8,"example1"}' > geneid_ucsc.gff
```

3c. To add the geneid GFF output press the Add custom tracks button in the UCSC genome browser. The geneid output modified in the previous step must be copied into the Data box. The following header must be placed just before the geneid GFF predictions:

```
##Including annotations into the UCSC genome browser
browser position chr21:13903812-13935812 track name
=geneid description="geneid predictions" visibility=1
color=0,150,50
```

A new track named "geneid" will be imported in the UCSC genome browser annotations. It is very important to maintain the tabular separation between columns. The user can select different displaying options (e.g., color, shape, etc.). The number and type of tracks to be displayed in the image can also be selected here, adding other commands in the header.

4c. Press the Go to the genome browser button to show the geneid GFF predictions incorporated into the UCSC genome browser annotations.

The main UCSC genome window will appear again. The geneid track is now displayed on top of the picture in green. Coding exons are represented as boxes. Exons belonging to the same gene are joined together by a line. Users can easily add or remove other sources of information. Figure 4.3.7 shows the UCSC display of the geneid prediction obtained in

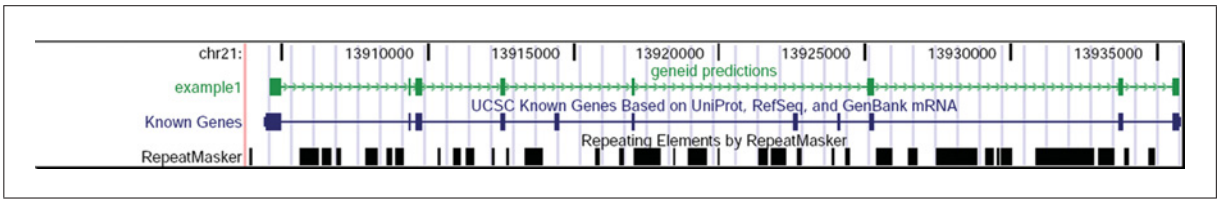


Figure 4.3.7 Using the UCSC genome browser to visualize geneid output.

Basic Protocol 1, step 6, the Known Genes track and the Repeat Masker results (see the Guidelines for Understanding Results for further discussion on gene prediction accuracy and sequence masking). The orientation of the arrows in the geneid track denoting the gene is annotated in the forward strand.

BASIC PROTOCOL 3

USING EXTERNAL INFORMATION TO SOLIDIFY geneid PREDICTIONS

One of the strengths of geneid is that it can easily incorporate external information about gene features on the input query sequence in the final gene prediction. As human genomic sequences are being annotated with increasing reliability, this option may be useful, e.g., to analyze in detail apparently void genomic regions lying between known genes, to explore the possibility of alternative exons in known genes with well established constitutive exonic structure, or to extend gene predictions based on partial EST sequences. This external evidence can include known exons, genes, or simply regions highly suspected of coding for proteins. In such cases, geneid will predict a gene structure compatible with the external information provided. The external information can also be a set of candidate exons obtained using some other exon-prediction approach (computational or experimental). In this case, geneid will assemble the gene prediction by maximizing the sum of the scores of the assembled exons. In any case, the gene features to be used by geneid as external information must be provided as GFF files.

The following describes two examples for which external information substantially improves geneid predictions.

Necessary Resources

Hardware

Unix/Linux workstation with at least 256 Mb RAM (recommended)

Software

geneid v1.2 full distribution (see Support Protocol)
Unix text editor

Files

This protocol uses the two human genomic sequences listed below which were extracted from the UCSC human genome browser (assembly March 2006). These sequences can be found at the samples subdirectory within the geneid distribution (see Support Protocol) and also at the *Current Protocols in Bioinformatics* Web site at *http://www.currentprotocols.com*.

`example2.fa` (47 kb):
Location: Human chromosome 22, coordinates 17,499,857-17,546,853 (reverse strand)
`example3.fa` (32 kb, masked):
Location: Human chromosome 15, coordinates 72,071,133-72,117,117
`example2.evidences.gff`, `example3.EST1.gff`, `example3.EST2.gff`, `example3.EST3.gff` and

`example3.promoter.gff` (contain annotated gene features on the above sequences)

1. Run geneid on the second example (`example2.fa`):

```
%geneid -P param/human3iso.param samples/example2.fa
```

geneid predicts a 21-exon gene on the forward strand. Figure 4.3.8, panel A, displays the default geneid prediction using gff2ps. The region actually encodes three different genes (all of them sharing exons with the geneid prediction). For the example, however, assume that at the time of the prediction only one of these genes, the second, has been determined. By providing the exonic structure of this gene, the overall geneid prediction in this region improves substantially.

2. Include external information in the geneid prediction by using the option `-R`:

```
%geneid -P param/human3iso.param -R
samples/example2.evidences.gff samples/example2.fa
```

Gene features (exons and genes) can be externally provided to geneid. The program then produces gene predictions that incorporate these features. These gene features are supplied in a GFF file. External gene features must be of a geneid exon type (First, Internal, Terminal, or Single; to work with partially supported exons, see Suggestions for Further Analysis). The strand on which they occur must also be provided, but frame and score are optional (by placing a "." in the GFF corresponding field). The GFF fields `seqname` *and* `source` *are not used, and they can be anything. Users should be aware, however, that if a score is specified for provided exons, these will compete with geneid-predicted exons, and may not be included in the final prediction. The group field in the GFF file can be used to prevent geneid from predicting additional exons within a known gene. Exons with the same group identifier are considered to belong to the same gene, and no additional exon is predicted between them (see geneid manual for details). External gene features are provided to geneid by means of the* `-R` *option, followed by the name of the GFF file. In the case of the example, the GFF file including the exon coordinates of the known gene is (remember that, in GFF, fields are delimited by tabs):*

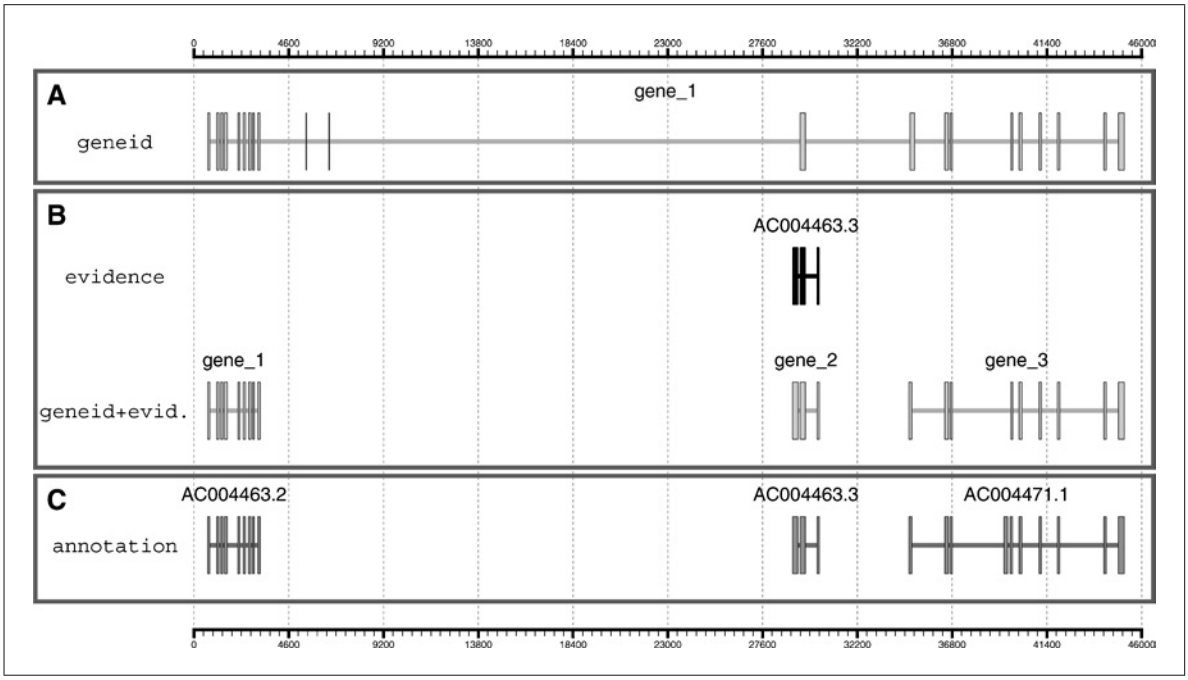


Figure 4.3.8 Improving gene prediction by using external information (Basic Protocol 3). (**A**) Default geneid prediction on sequence example2. (**B**) geneid prediction when the exon coordinates of gene AC004463.3 are given to geneid. (**C**) Ensembl annotation of the sequence.

```
example2 known_gene    First 29058 29316 . + . AC004463.3
example2 known_gene Internal 29425 29678 . + . AC004463.3
example2 known_gene Terminal 30246 30350 . + . AC004463.3
```

Since we are assuming that the exonic structure of the second gene is completely determined, all the exons in the GFF file must share the same group identifier. The new prediction obtained by geneid appears in Figure 4.3.8, panel B. This prediction is now very similar to the actual gene structure in this region of the human genome.

Using external information to investigate alternative splicing forms with geneid

3. Run geneid on the third example (`example3.fa`):

```
%geneid -P param/human3iso.param samples/example3.fa
```

geneid predicts a six-exon gene in the forward strand (see Fig. 4.3.9, panel A). It is known that this gene has a number of splice isoforms (Fagioli et al., 1992), some of them being

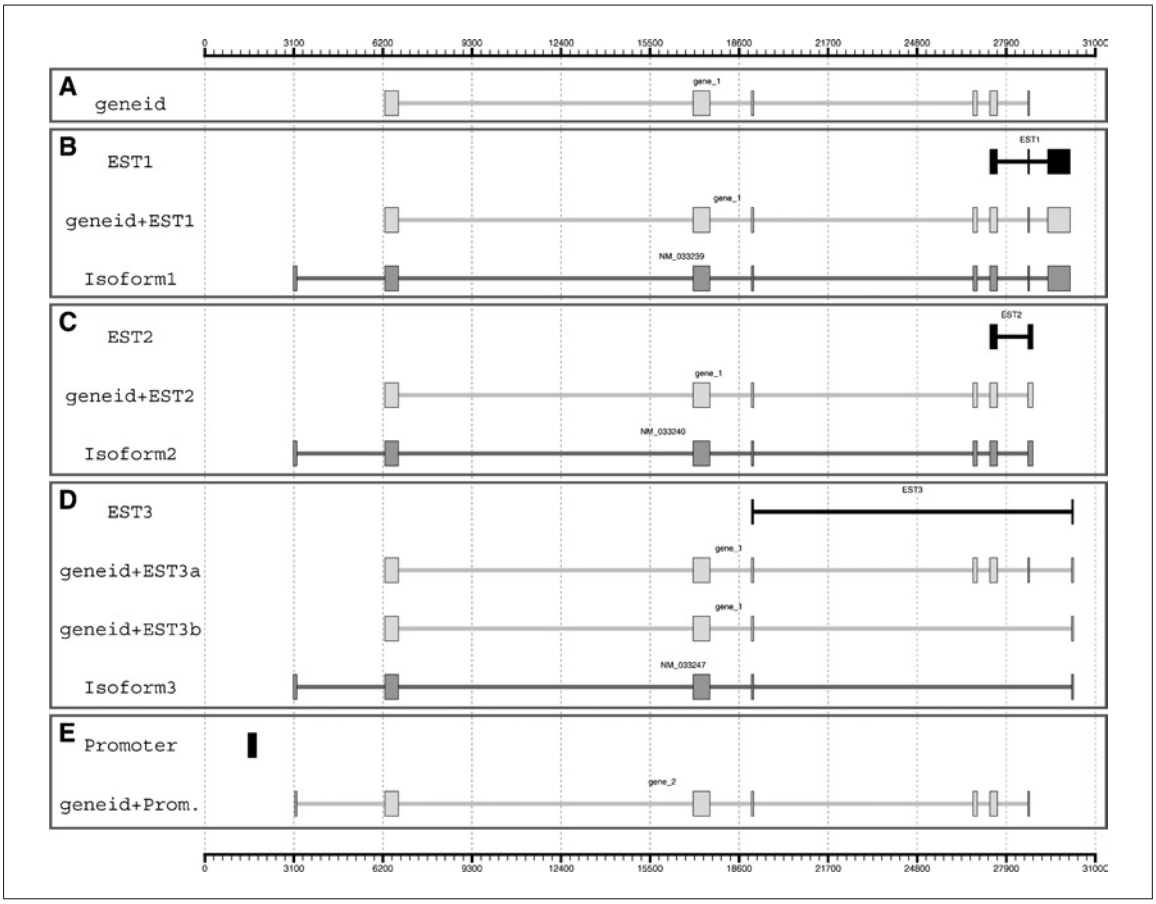


Figure 4.3.9 Using external information to investigate alternative splicing forms with geneid (Basic Protocol 3). (**A**) Default geneid prediction on sequence example3. (**B, C**) Prediction of two alternative transcripts. The EST1 and EST2 tracks display the exonic structure of partial ESTs matches whose coordinates have been given to geneid. geneid+EST1 and geneid+EST2 show the resulting geneid predictions. Isoform1 and Isoform2 correspond to the coordinates of the two isoforms. (**D**) Prediction of a third alternative transcript. The EST3 track displays the exonic structure of the EST, whose genomic coordinates has been given to geneid. geneid+EST3a and geneid+EST3b display the geneid predictions before and after the exon filtering process. The Isoform3 track contains the annotation for this isoform. (**E**) The coordinates of a promoter element (Promoter; may be obtained by experimental means) are given to geneid, which improves the prediction of the first coding exon (geneid+Prom).

displayed in Figure 4.3.9, panels C, D, and E. Assume, however, that these isoforms are unknown. Since a number of ESTs align to this genomic region, supporting alternative 3′-end exonic structures, this example will show how geneid can be used to extend these EST alignments to recover the full alternative transcript in each case.

4. Use geneid to extend a gene structure derived from a given EST. Type:

```
%geneid -P param/human3iso.param -R
samples/example3.EST1.gff samples/example3.fa
```

The genomic coordinates of the alignment of one of these ESTs (EST1) to the genomic sequence—obtained, e.g., using ESTgenome (Mott, 1997), GeneWise (Birney and Durbin, 2000), or any other cDNA-to-genomic-DNA alignment tool—are included in a GFF file which is passed via the `-R` *option into geneid. These programs obtain a so-called spliced alignment between the EST sequence and the genomic query. In such an alignment, big gaps—likely to correspond to introns—are only allowed at legal splice junctions. The GFF file in this case is:*

```
example3 EST1 Internal 27330 27588 . + .
example3 EST1 Internal 28652 28704 . + .
example3 EST1 Terminal 29345 30124 . + .
```

The result of the prediction appears in Figure 4.3.9B. geneid predicts a product (distinct from the default prediction) which incorporates the three exons in the EST sequence, and which resembles closely one of the known alternative forms for this gene.

5. Use geneid to obtain an alternative structure supported by a different EST:

```
%geneid -P param/human3iso.param -R
samples/example3.EST2.gff samples/example3.fa
```

The genomic coordinates of the alignment of a different EST (EST2) are given now to geneid. The corresponding GFF file is:

```
example3  EST2  Internal  27330  27588  .  +  .
example3  EST2  Terminal  28652  28830  .  +  .
```

The prediction incorporates the two exons in the EST sequence (Figure 4.3.9C), and resembles closely another of the known alternative forms for this gene.

6. Use geneid to obtain an alternative structure supported by a different EST:

```
%geneid -P param/human3iso.param -R
samples/example3.EST3.gff samples/example3.fa
```

The genomic coordinates of the alignment of a different EST (EST3) are given now to geneid. The corresponding GFF file is:

```
example3  EST3  Internal  19031  19101  .  +  .
example3  EST3  Terminal  30180  30233  .  +  .
```

The resulting prediction appears in Figure 4.3.9D (geneid + EST3a). As it is possible to see, the geneid predictions include new exons between the two exons corresponding to the EST sequence. The resulting prediction is thus incompatible with the EST sequence. Grouping the EST sequences into a gene would certainly prevent the inclusion of these exons. In the current version of geneid, however, grouped features cannot be extended. Although the procedure is somehow more complex, it will also serve to illustrate the option `-O`*, which allows geneid to produce gene predictions from sets of exons provided externally. Essentially, the user must predict the exhaustive list of exons along the genomic sequences, eliminate (knock out) those exons occurring between the two EST3 matches, and run geneid from the remaining set of exons.*

7. Predict all exons on sequence example3 (`example3.fa`):

```
%geneid -P param/human3iso.param -xoGP
samples/example3.fa > example3.exons.gff
```

Option `-x` instructs geneid to print all exons, option `-o` forces geneid to switch off gene prediction, and option `-G` produces GFF output.

8. Open the file `example3.exons.gff` with a text editor. First, discard all predicted exons between the two exons supported by EST3, i.e., those in the range from the position 19,031 to the position 30,233. Then open `example3.EST3.gff` and add the content at the end of the file. Save the new file as `example3.filtered.exons.gff` and close the editor. Finally, use the Unix command `sort` on this file to obtain the ordered list of exons, type:

```
%sort +3n example3.filtered.exons.gff
```

This operation can be also accomplished using a number of Unix file-editing tools (such as awk). For instance, the awk command would be:

```
%awk '$5<19031||$4>30233'example3.exons.gff | cat -
  samples/example3.EST3.gff | sort +3n >
  example3.filtered.exons.gff
```

The coordinates of the known EST have to be included in the file of candidate exons, because in geneid v1.2, the `-R` and `-O` options are incompatible (see next step).

9. Predict the gene structure in sequence example3 from the set of remaining exons:

```
%geneid -P param/human3iso.param -O
example3.filtered.exons samples/example3.fa.
```

Option `-O` instructs geneid to read the set of predicted exons externally, instead of predicting them, and assemble the optimal gene structure from this set. The resulting prediction appears in Figure 4.3.9D (geneid + EST3b), which is now compatible with the EST3 sequence and which closely resembles yet another isoform.

10. Force the prediction of the first exon of the gene by providing the coordinates of the promoter element:

```
%geneid -P param/human3iso.param -R
samples/example3.promoter.gff samples/example3.fa >
example3.exons.gff
```

Even though geneid predictions on example3 correspond quite well to different isoforms of the same gene, in all cases geneid fails to predict the first coding exon of the gene. Failing to predict short first coding exons is a "feature" of geneid, as well as of other gene prediction programs. With geneid, there are a number of ways in which the user can force the prediction of a complete gene (starting by a First exon)—e.g., by using a gene model, which defines (see Background Information) how to assemble only one gene. This examples uses the fact that the default gene model includes a promoter feature (see Background Information) to provide geneid the coordinates of a promoter element, which has been experimentally determined. Such coordinates of the "promoter" element are given in the GFF file `samples/example3.promoter.gff`:

```
example3  experimental  Promoter  1500  1799  .  +  .
```

The prediction includes now a first coding exon, similar to the annotated one (Fig. 4.3.9E).

USING THE geneid WEB SERVER TO PREDICT GENES

ALTERNATE PROTOCOL

A Web interface to geneid can be accessed at *http://genome.imim.es/geneid.html*. The geneid server consists of a form to input the DNA sequence (which is mandatory) as well as the external information to improve the prediction (which is optional), providing a set of different options to customize the behavior of the program. All of the geneid functionality is available through the geneid Web server, in particular the operations and commands described in the previous protocols (see Basic Protocol 1 and 3). Moreover, this server can supply a graphical representation of the predictions obtained with the program gff2ps (see Basic Protocol 2). This protocol outlines the use of this interface to predict genes as well as other genomic elements on DNA sequences.

The geneid Web server is divided basically into three main areas according to the type of information they provide to the user: Input Data (Fig. 4.3.10), Prediction Options (Fig. 4.3.11), and Output options (Fig. 4.3.12). Once the user has supplied a sequence to process and selected the appropriate options, the form containing this information must be transferred from the user client to the geneid server by clicking on the button "Submit." Depending on the complexity of the query and the length of the input sequence, the results (Fig. 4.3.13) will be returned to the user in a reasonably short period of time. The form can be reset and its content deleted with the button "Reset form." Users can obtain help through several links in the Web page.

Necessary Resources

Hardware

A computer and a connection to the Internet

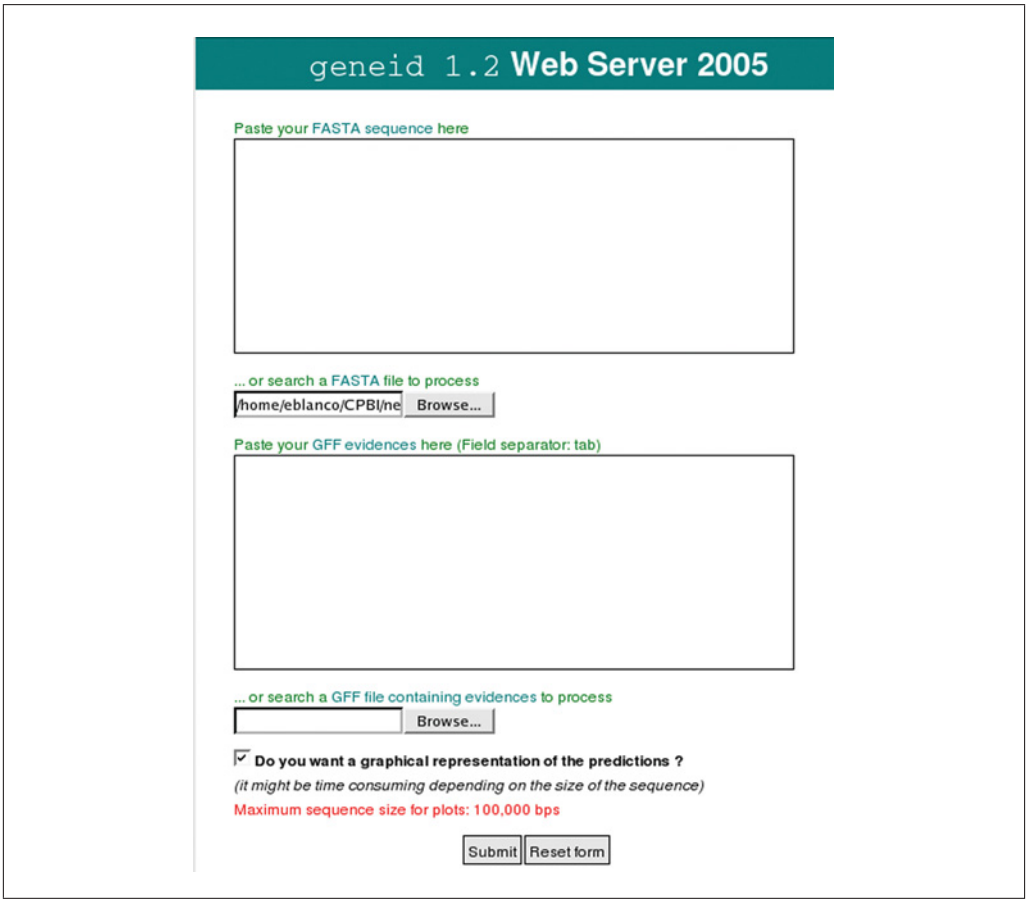

geneid 1.2 Web Server 2005
Paste your FASTA sequence here
... or search a FASTA file to process
/home/eblanco/CPBI/ne
Browse...
Paste your GFF evidences here (Field separator: tab)
... or search a GFF file containing evidences to process
Browse...
Do you want a graphical representation of the predictions ?
(it might be time consuming depending on the size of the sequence)
Maximum sequence size for plots: 100,000 bps
Submit
Reset form

Figure 4.3.10 geneid Web server: DNA and external information area.

Software

An up-to-date Internet browser, such as Internet Explorer (*http://www.microsoft.com/ie*); Netscape (*http://browser.netscape.com*); Firefox (*http://www.mozilla.org/firefox*); or Safari (*http://www.apple.com/safari*).

Files

All of the sequences (in FASTA format; *APPENDIX 1B*) and external information used in Basic Protocol 1 and 3

1. Input DNA sequence and external information using the Input Data section (Fig. 4.3.10).

   This section contains two text areas: one to copy/paste the DNA sequence and the other for the external information to improve the predictions (see Basic Protocol 3), as well as a button to select a graphical representation of the results.

   Users must input a DNA sequence in FASTA format (APPENDIX 1B) either from file or from the text area, while the external information in GFF format is optional. The process for building a graphical representation from the geneid output with the program gff2ps can be time-consuming when the length of the input sequence is more than 100 kbp, and the geneid server might dismiss the query to prevent overloading the server. By default, this option is disabled.

2. Configure Prediction Options (Fig. 4.3.11).

   There are three different geneid features to configure: the organism, the mode, and the DNA strands to be scanned for genes. All of these fields share the same structure: a set of possible values from which the user can only select one.

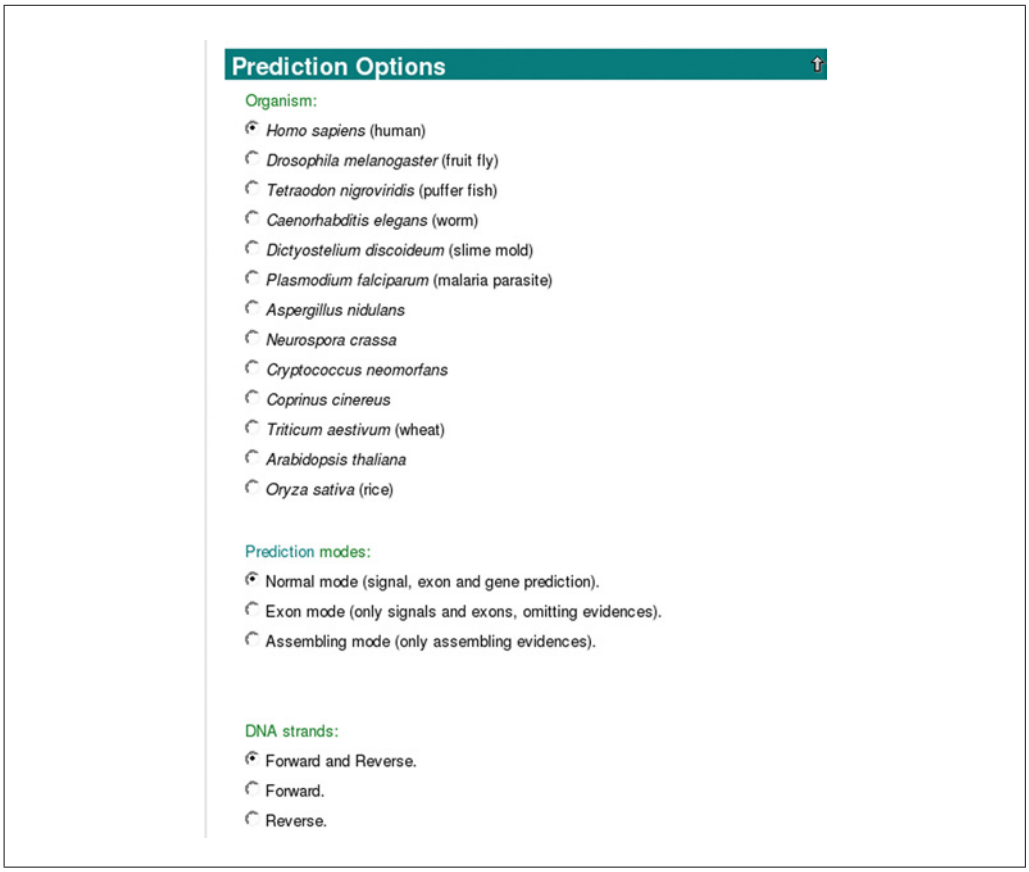


Figure 4.3.11 geneid Web server: Prediction Options area.

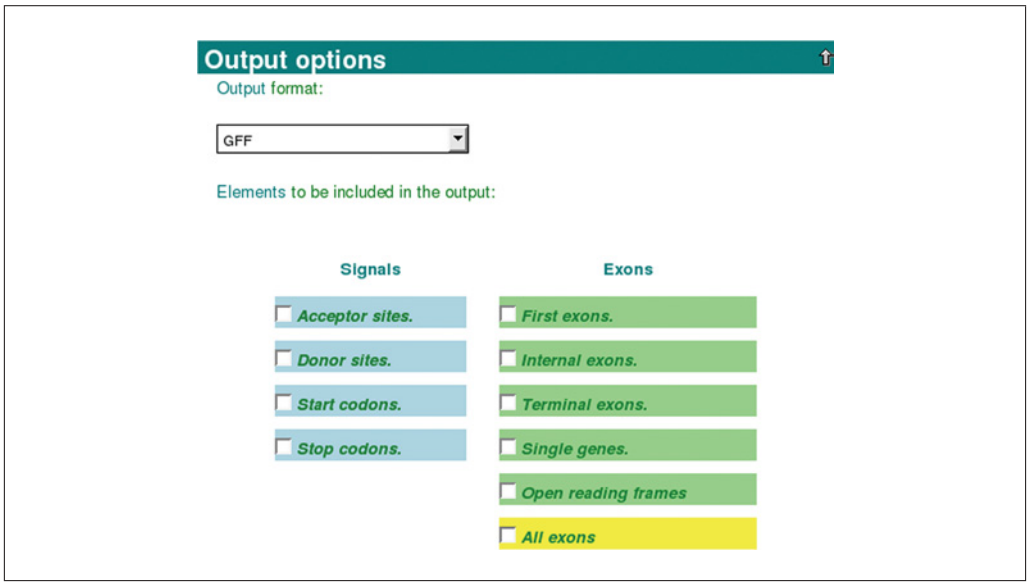


Figure 4.3.12 geneid Web server: Output Options area.

In the Organism menu, users will select the suitable organism depending on the species that the DNA sequence is from (see Guidelines for Understanding Results, The Parameter File). Currently, the available organisms are Homo sapiens (default), Drosophila melanogaster, Tetraodon nigroviridis, Caenorhabditis elegans, Dictyostelium discoideum, Plasmodium falciparum, Aspergillus nidulans, Neurospora crassa, Cryptococcus neomorfans, Coprinus cinereus, Triticum aestivum, Arabidopsis thaliana and Oryza sativa.

In the Prediction Modes menu, the geneid engine can be configured to predict either signals, exons, or genes. Depending on the input information, users will select "Normal mode" to obtain the optimal genes predicted on the sequence (see Basic Protocol 1, steps 1 and 2) or to reannotate the current sequence by using external information, if provided (see Basic Protocol 3), "Exon mode" to predict only signals and exons, disabling gene assembling (see Basic Protocol 1, steps 3 and 4), or "Assembling mode" to only assemble the best genes from the external information, when provided (e.g. predictions from gene prediction programs other than geneid, in GFF format).

In the DNA Strands menu, the user can select where to predict genomic elements: "Forward and Reverse" (default), "Forward" (positive), or "Reverse" (negative).

3. Choose the output format and elements to be displayed in the Output Options section (Fig. 4.3.12).

There are two different sets of Output Options: those concerning the format and those concerning the elements to display. The available formats are GFF, geneid, extended format, and XML, as well as a format containing the CDS sequence for each predicted gene (for further details about the formats see Basic Protocol, steps 2, 5, 6, and 7).

The signals that can be included in the output are Acceptor and Donor splice sites and Start and Stop codons. There are five types of exons: First, Internal, Terminal, Single, and ORFs. There is also an option to build an ordered output containing all of the predicted exons (see Basic Protocol 1, step 2 for details about the type of genomic elements predicted by geneid).

4. Examine the geneid output (Fig. 4.3.13).

The results for the sequence `example1.fa` *(see Basic Protocol 1 steps 1 and 2, for detailed explanation) are shown in Figure 4.3.13. The input file (DNA sequence) was given in the Input Data section (see Fig. 4.3.10) and the option Do You Want a Graphical Representation of the Predictions in the Input Data section was checked. In the Prediction Options section (see Fig. 4.3.11), "Homo sapiens" was selected for Organism, "Normal mode" was selected for Prediction Mode, and "Forward and Reverse," was selected for DNA Strands. In the Output Options section "GFF" was selected for Output Format.*

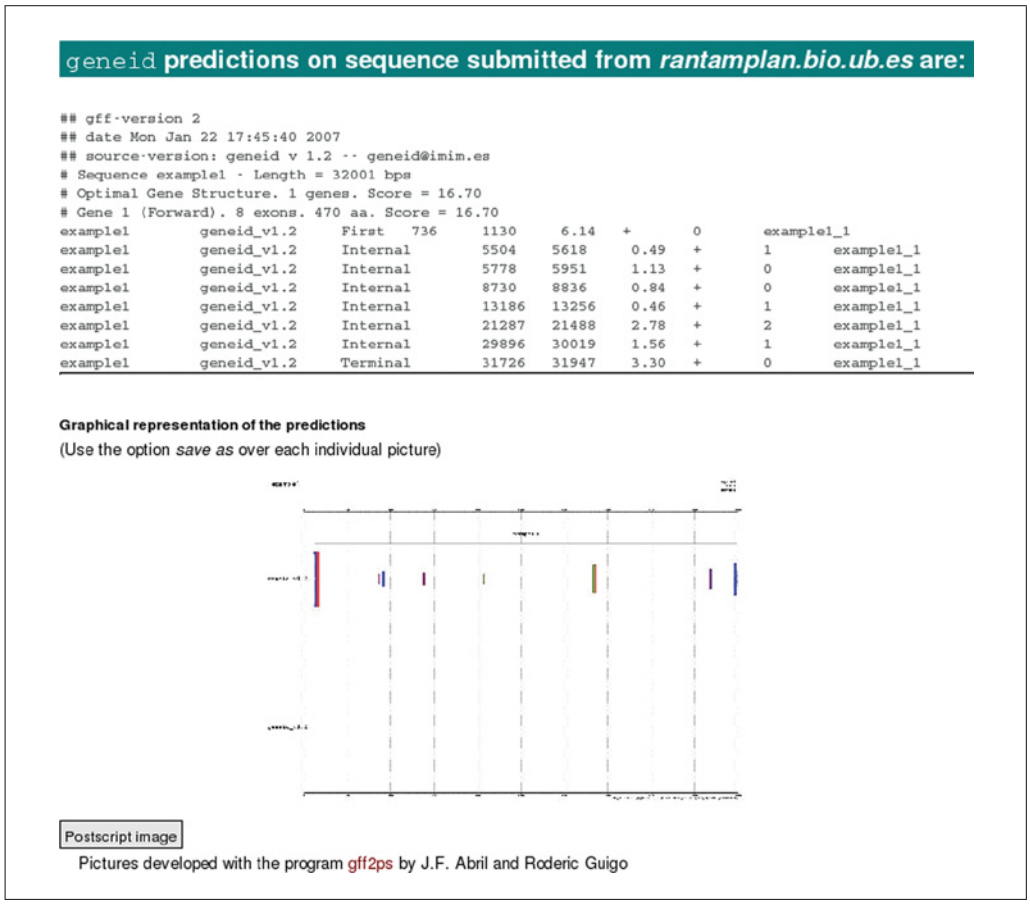

geneid predictions on sequence submitted from rantamplan.bio.ub.es are:
gff-version 2
date Mon Jan 22 17:45:40 2007
source-version: geneid v 1.2 -- geneid@imim.es
Sequence example1 - Length = 32001 bps
Optimal Gene Structure. 1 genes. Score = 16.70
Gene 1 (Forward). 8 exons. 470 aa. Score = 16.70
example1 geneid_v1.2 First 736 1130 6.14 + 0 example1_1
example1 geneid_v1.2 Internal 5504 5618 0.49 + 1 example1_1
example1 geneid_v1.2 Internal 5778 5951 1.13 + 0 example1_1
example1 geneid_v1.2 Internal 8730 8836 0.84 + 0 example1_1
example1 geneid_v1.2 Internal 13186 13256 0.46 + 1 example1_1
example1 geneid_v1.2 Internal 21287 21488 2.78 + 2 example1_1
example1 geneid_v1.2 Internal 29896 30019 1.56 + 1 example1_1
example1 geneid_v1.2 Terminal 31726 31947 3.30 + 0 example1_1
Graphical representation of the predictions
(Use the option save as over each individual picture)
Postscript image
Pictures developed with the program gff2ps by J.F. Abril and Roderic Guigo

Figure 4.3.13 geneid Web server output with the sequence example1.fa.

The output is divided into two main areas: the plain text output (see Basic Protocol 1, step 6) and the graphical output of the predictions (see Basic Protocol 2, step 1a). Images are provided in JPG format, although a PostScript document can be generated on the fly by switching the Postscript Image button on. At the bottom of the output, information about the process, parameters used, and options is displayed.

SUPPORT PROTOCOL

HOW TO GET geneid AND VISUALIZATION PROGRAMS

The geneid Web page is at *http://genome.imim.es/software/geneid/index.html*. From this page, users can download the software and the accompanying documentation, and obtain other information about geneid. The geneid software can also be downloaded by anonymous FTP from *ftp://genome.imim.es*. The program geneid is written in ANSI C and runs on Unix-based operating systems such as Linux, MacOSX, Solaris, and Irix. geneid source code, compiled binaries, parameter files, and documentation are available under the GNU GENERAL PUBLIC LICENSE. This protocol describes how to download and install geneid.

Necessary Resources

Hardware

Unix/Linux workstation with at least 256 Mb RAM (recommended), and Internet access.

Software

An up-to-date Internet browser, such as Internet Explorer (*http://www.microsoft.com/ie*); Netscape (*http://browser.netscape.com*); Firefox (*http://www.mozilla.org/firefox*); or Safari (*http://www.apple.com/safari*).

Files

None

To obtain the geneid software from the Web, go to the Distribution section of the geneid Web page, and click on Full Distribution (Current release: geneid v1.2).

To obtain the geneid software by anonymous FTP, run an FTP session as follows:

```
%ftp genome.imim.es
Name: anonymous
Password: -
ftp> cd /pub/software/geneid
ftp> binary
ftp> get README
ftp> get geneid_v1.2.March_1_2005.tar.gz
ftp> quit
```

The geneid distribution has been compressed in a single file `geneid_v1.2.March_1_2005.tar.gz`, using the Linux command `tar -zcvf`. To uncompress and extract the files, type the following commands:

```
%gzip -d geneid_v1.2.March_1_2005.tar.gz
%tar -xvf geneid_v1.2.March_1_2005.tar
```

On Linux systems, type:

```
%tar -zxvf geneid_v1.2.March_1_2005.tar.gz
```

After uncompressing the geneid distribution, the directory `geneid/` will have been created in the current working directory. The `geneid` directory contains several subdirectories and files

`docs/`: geneid documentation
`include/`: geneid header file
`param/`: Parameter files for several organisms
`samples/`: FASTA sequences used in this unit
`src/`: geneid source code
`GNU License`
`Makefile`: Makefile to build the binary file
`README`

Before starting to work with geneid, it is necessary to compile the program, i.e., produce a binary file properly generated according to the computer architecture. For that, move to the geneid directory by typing:

```
%cd geneid
```

To compile the program building a binary file, type:

```
%make
```

A new directory called `bin/` has been created now. Inside, the geneid program is ready to be executed by the users. Just to test the program showing the list of available options, try the command:

```
%bin/geneid -h
```

On most Unix systems, this should be fairly simple but if you encounter problems, please contact the authors at *geneid@imim.es*. Throughout this unit, for simplicity, the relative path `bin/` has been omitted in the examples, just running `%geneid`. It is also advisable to set the `GENEID` environmental variable to point to the `param/` subdirectory within the `geneid` directory.

The geneid distribution includes complete and exhaustive documentation. It has been written in HTML, and it can be accessed through a Web browser. The documentation is also available at the geneid homepage.

How to get gff2ps

The gff2ps Web page contains the information required to download and install the program (*http://genome.imim.es/software/gfftools/GFF2PS.html*). gff2ps can also be downloaded by anonymous FTP from *ftp://genome.imim.es/pub/software/gff_tools/gff2ps/*.

How to obtain apollo

The apollo Web page contains the information required to download and install the program: *http://www.fruitfly.org/annot/apollo/*

The default apollo configuration deals with geneid GFF files successfully. However, to reproduce the graphical representation in Figure 4.3.6, one will need to add some lines to the file `ensj.tiers`. This file contains the display specifications for different gene features. Therefore, open the file with a Unix editor such as pico, joe, or emacs, type the following lines at the end of the file, save the resulting file, and close the editor:

```
[Tier]
tiername : geneid_v1.2
[Type]
label : geneid_v1.2
tiername : geneid_v1.2
datatype : geneid_v1.2
glyph : DrawableGeneFeatureSet
color : 255,204,0
column : GENOMIC_LENGTH
column : GENOMIC_RANGE
column : SCORE
```

In addition, the style attributes that must be set in the preferences file are:

```
FeatureBackgroundColor        "black"
EdgematchColor                "green"
EdgematchWidth                "1"
CoordBackgroundColor          "black"
CoordForegroundColor          "yellow"
Draw3D                        "true"
```

GUIDELINES FOR UNDERSTANDING RESULTS

Despite significant advances in the field of computational gene prediction, current gene-finding methods are far from being able to accurately predict the exonic structure of the genes encoded in large genomic sequences (for a recent exhaustive evaluation of gene-prediction programs in a subset of the human genome sequence, see Guigó et al., 2006). Although a large fraction of the existing genes will be at least partially predicted by existing tools, only a small fraction will be predicted in a completely correct fashion.

On the other hand, gene finders tend to overpredict genes, resulting in a large number of false-positive gene predictions. Current methods deal poorly with not so uncommon phenomena such as alternative splicing, genes with unusual codon composition, nested genes, genes within introns, noncanonical splice sites, and exceptions to the standard genetic code (such as those characterizing the selenoproteins). Gene boundaries are also poorly predicted, often resulting in split or chimeric gene predictions. All these drawbacks need to be taken into consideration when interpreting the results of gene-prediction programs, not only those of geneid (see Zhang, 2002 for a review). The following discusses some more specific features of geneid.

Accuracy of Geneid: Specificity Versus Sensitivity

As discussed above, most gene finders suffer from lack of specificity, predicting a large number of false-positive exons and genes, particularly in large genomic sequences. The authors believe that, comparatively, geneid has superior specificity to other existing gene finders, showing a somewhat more conservative behavior. The price is paid in terms of sensitivity. geneid v1.2 may miss more real exons than other gene finders. This is particularly true for short exons. Compared to other programs, the problem is more relevant when analyzing single gene sequences. The coding fraction of initial exons is often very short, and geneid may not resolve it well, missing it completely or extending it into a longer internal exon. When analyzing sequences coding for only one gene, the authors recommend that a gene model (see below) be used, which forces the prediction of a single gene in the query genomic sequence. This single gene can also be forced to be complete, thus necessarily starting with a first exon and ending with a terminal exon (see geneid manual). Exhaustive analysis (data not shown) indicates that when using this option, the accuracy of geneid predictions in single gene sequences compares favorably to that of other gene finders. In general, for large genomic sequences encoding multiple genes, the overall accuracy of default geneid is comparable, if not superior, to that of the most accurate existing tools, offering a better balance between specificity and sensitivity (see the geneid Web page for a discussion of accuracy).

Gene and Exon Scores

Gene and exon scores have a probabilistic interpretation within geneid (see Background Information). Thus, although the authors have not studied exhaustively the false-positive rate of exon predictions as a function of the score, as a rule of thumb, the higher the score of an exon, the higher its likelihood. Note, however, that in geneid the score of an exon depends directly on its length, and that a very short exon cannot, by definition, have a high score. Thus, very short exons may have very low, even negative, scores.

UTRs

geneid, as with most genefinders, predicts only the coding fraction of a gene. Usually, users are interested mainly in the gene protein product, and this is not an important limitation. However, untranslated exons may contain good splice signals, and although their nucleotide composition does not reflect the codon bias characteristic of protein coding regions, they appear to exhibit a higher nonrandom bias than intronic or intergenic DNA. It is, thus possible that, in some cases, geneid predictions may include portions of a gene UTR.

Masking the Sequence

Some types of interspersed repeats and low-complexity regions exhibit a highly nonrandom sequence composition, often similar to that characterizing protein coding regions

(Stormo, 2000). geneid may include these in the gene predictions. It may be advisable, thus, to mask the query sequence for such repeats and regions using, for instance, the program RepeatMasker (*http://repeatmasker.org*) before running geneid. This strategy may increase the specificity of the predictions. Let us note, however, that real genes often include low-complexity regions. Extreme masking of the query sequence may lead to some genes (or fraction of genes) being missed.

G+C Content

Accuracy of predictions may be quite sensitive to G+C content. Indeed, gene structure has been reported to depend on the G+C content. However, different programs appear to behave differently with respect to G+C content. In general, geneid predictions are poorer in low G+C content sequences.

The Parameter File

geneid needs a parameter file to build the predictions. This parameter file is computed explicitly for a given species or taxonomic group. Currently, there are parameter files for *Homo sapiens* (which can be safely applied to all mammalian sequences), *Tetraodon nigroviridis* (which can be safely used, at least in other pufferfish species, such as *Fugu rubripes*), *Drosophila melanogaster* (probably extensible to other diptera species), *Caenorhabditis elegans, Dictyostelium discoideum, Solanum lycopersicum, Triticum aestivum, Oryza sativa, Arabidopsis thaliana*, and many others. New models are regularly uploaded at the geneid Web page.

The parameter file contains mostly the description of the probabilistic model on which the predictions are based (see Background Information): Position Weight Matrices (PWM) to predict sites, and the Markov model to score candidate exons. These need to be estimated from large training sets of sequences, and users, in general, are not expected to modify them. However, to reduce computation time and memory required, geneid uses a number of cutoffs to further consider predicted sites and exons. In some cases, users may want to modify these cutoffs to increase or decrease the size of the set of candidate exons and sites. For instance, users may want to predict and score every GT dinucleotide as a candidate donor site. In such a case, the cutoff associated with the PWM for donor sites should be set to a very low number (–99, for instance). See the geneid manual for details. For same species, parameters are specifically estimated for regions with different G+C content (isochores).

The Gene Model

From a large number of candidate exons, geneid selects a proper combination of exons to assemble the predicted gene structure. This assembly must conform to a number of biological constraints, for example, that selected exons cannot overlap, or that an Open Reading Frame (ORF) should be maintained along the assembled gene.

These biological constraints are defined in a set of rules in the so called gene model, included within the parameter file. These rules refer to the order of gene features in the prediction and to the distances between them. Each rule is a three column record in the gene model. For instance, the rule:

```
First+:Internal+  Internal+:Terminal+  40:10000
```

indicates that elements (exons) of type Internal in the forward strand and of type Terminal in the forward strand are allowed only immediately after exons of type First in the forward strand or of type Internal in the forward strand. The third column indicates the valid

```
#Gene_Model
#intronic connections
First+:Internal+                Internal+:Terminal+              20:40000 block
Terminal-:Internal-             First-:Internal-                 20:40000 blockr
#connections to regulatory elements
Promoter+                       First+:Single+                   50:4000
Terminal+:Single+               aataaa+                          50:4000
First-:Single-                  Promoter-                        50:4000
aataaa-                         Single-:Terminal-                50:4000
#intergenic connections
aataaa+:Terminal+:Single+       Single+:First+:Promoter+         500:Infinity
aataaa+:Terminal+:Single+       Single-:Terminal-:aataaa-        500:Infinity
Promoter-:First-:Single-        Single+:First+:Promoter+         500:Infinity
Promoter-:First-:Single-        Single-:Terminal-:aataaa-        500:Infinity
```

Figure 4.3.14 geneid Default Gene Model.

distances at which these elements can be assembled into a predicted gene. In this case, these elements must be at least 40 bp and at most 10,000 bp apart. Note that this rule specifies the constraints governing intronic connections in the forward strand. The basic gene model distributed with geneid v1.2 appears in Figure 4.3.14.

Note that the default gene model includes rules for promoter elements and poly(A) signals. The current version of geneid, however, predicts only elements of type First, Internal, Terminal, or Single. Predicted promoter elements or poly(A) signals (probably obtained using other programs), must be passed as external information via the `-R` option (see Basic Protocol 3). Users can modify the gene model to consider other features, but the predicted features must be passed to geneid also via the `-R` option. Modification of the gene model may not involve the introduction of new features, but changing the rules affecting default features, for instance to force the prediction of only one gene.

COMMENTARY

Background Information

History

The program geneid (Guigó et al, 1992) was one of the first programs to predict full exonic structures of vertebrate genes in anonymous DNA sequences. geneid was designed following a simple hierarchical structure: first, gene-defining signals were predicted and scored using weight matrices. Next, potential exons were constructed from these sites, and their coding potential was scored as a function of several coding statistics, such as hexamer composition, whose coefficients were estimated by a neural network. Finally the optimal-scoring gene prediction was assembled from the best exons by performing an exhaustive search of the space of possible gene assemblies, ranked according to a score obtained through a complex function of the score of the assembled exons.

Roderic Guigó, Steen Knudsen, and Neil Drake, in the Temple F. Smith group, contributed to this first version of geneid. This version was developed at the Molecular Biology Computer Research Resource (Dana Farber Cancer Institute, Harvard University). It was never distributed, but an E-mail server was set up in late 1991, which was latter moved to the Biomolecular Engineering Research Center (Boston University). Kathleen Klose and Steen Knudsen developed the server. In 1995, a Web server was set up at the Institut Municipal d'Investigació Mèdica (IMIM) in Barcelona. Moisès Burset developed the server.

Version 1.0 of geneid (Parra et al., 2000) was completely rewritten at the IMIM. This version maintained the hierarchical structure (signal to exon to gene) in the original geneid but the scoring schema was simplified and furnished with a probabilistic meaning, as discussed above. A new version of geneid (v1.1) was released in 2002. This version had a substantially improved engineering design which makes it more robust, faster, and more

memory-efficient. It was more accurate than version v1.0, parameter files were developed for a larger number of species, and more extensive documentation was supplied. The latest version of geneid (v1.2) has been released in 2004. Additional parameter files and other new features were implemented over the platform of the previous version to support comparative gene prediction. The code in the geneid v1.x versions is mostly by Enrique Blanco and Roderic Guigó, with contributions from Moises Burset and Xavier Messeguer. The parameter files for different species have been developed by Genís Parra and Francisco Camara. The current Web server has been written by Enrique Blanco at the IMIM.

geneid uses Position Weight Matrices (PWM) to predict potential splice sites and start codons. Potential sites are scored as log-likelihood ratios. From the set of predicted sites (which includes, in addition, all potential stop codons), the set is built of all potential exons. Exons are scored as the sum of the scores of the defining sites, plus the log-likelihood ratio of the Markov model for coding sequences. Finally, the gene structure is assembled from the set of predicted exons, maximizing the sum of the scores of the assembled exons.

Predicting and scoring sites

PWMs are used to score each potential donor site (GT), acceptor site (AG), and start codon (ATG) along a given sequence. The score of a potential donor site (if assumed to be of length l) $S = s_1 s_2 \ldots s_l$ within the sequence is computed as:

$$L_{\mathrm{D}}(S) = \sum_{j=1}^{l} D_{sij}$$

This is the log-likelihood ratio of the sequence S in an actual site versus S in any false GT site. D_{ij} is the logarithm of the ratio of the probability of nucleotide i in position j in an actual donor site over the probability of i in position j in a false site. D_{ij} values are estimated from a (training) set of positive and false donor sites. Similar scores are computed for Acceptor Sites (L_A) and Start Codons (L_B).

Predicting and scoring exons

geneid distinguishes four types of "exons":

Initial: ORFs defined by a start codon and a donor site.

Internal: ORFs defined by an acceptor site and a donor site.

Terminal: ORFs defined by an acceptor site and a stop codon.

Single: ORFs defined by an start codon and a stop codon. This corresponds to intronless genes.

geneid constructs all potential exons which are compatible with the predicted sites.

Coding potential: geneid uses a Markov model of order five to compute the likelihood of an exon sequence to be coding. The model is estimated from both exon and intron sequences. The probability distribution of each nucleotide, given the pentanucleotide preceding it, is estimated in a set of known exon and intron sequences. From the exon sequences, this probability is estimated for each of the three possible frames, and three transition probability matrices F^1, F^2, and F^3 are computed. F^j $(s_1 s_2 s_3 s_4 s_5 s_6)$ is the observed probability of finding hexamer $s_1 s_2 s_3 s_4 s_5 s_6$ with s_1 in codon position j, given that pentamer $s_1 s_2 s_3 s_4 s_5$ is with s_1 in codon position j. An initial probability matrix, I^j, is estimated from the observed pentamer frequencies at each codon position. From the intron sequences, a single transition matrix is computed, F_0, as well as a single initial probability matrix, I_0. Then, for each hexamer h and frame j a log-likelihood ratio is computed:

$$LF^j(h) = \log \frac{F^j(h)}{F_0(h)}$$

as well as for each pentamer p and frame j:

$$LI^j(p) = \log \frac{I^j(h)}{I_0(h)}$$

Then, given a sequence S of length l in frame j, the coding potential of the sequence is defined as:

$$L_{\mathrm{M}}(S) = LI^j(S_{1\ldots5}) + \sum_{i=1}^{l-5} LF^j(S_{1\ldots i+5})$$

where $S_i \ldots_j$ is the subsequence of S starting in position i and ending in position j.

The score of a potential exon, S, $L_{\mathrm{E}}(S)$ defined by sites s_{a} (start/acceptor) and s_{d} (donor/stop) is computed as the following log-likelihood score:

$$L_{\mathrm{E}}(S) = L_{\mathrm{A}}(s_{\mathrm{a}}) + L_{\mathrm{D}} + (s_{\mathrm{d}}) + L_{\mathrm{M}}(S)$$

Assembling genes

geneid predicts gene structures (which can be multiple genes in both strands) as sequences of frame-compatible nonoverlapping exons. If

a gene structure g is a sequence of exons, e_1, $e_2 \ldots e_n$, the score of the gene is the log-ratio:

$$L_G(g) = L_E(e_1) + L_E(e_2) + \ldots + L_E(e_n)$$

In geneid, the gene structure predicted for a given sequence is the gene-maximizing $L_G(g)$, among all those gene structures that can be assembled from the set of predicted exons. An efficient dynamic programming algorithm is used to find the gene structure G, maximizing L_G (Guigó, 1998). Actually, because of a number of approximations made, the simple sum of log-likelihood ratios does not produce necessarily genes with the "right" number of exons (if L_E tends to be positive, the genes tend to have a large number of exons; if L_E tends to be negative, the genes tend to have an small number of exons), and the score of the exons is corrected by adding a constant EW. Thus, given an exon e, the actual score of e is:

$$L_E{}^*(e) = L_E(e) + \mathrm{EW}$$

To estimate this constant, a simple optimization procedure is performed. The value of EW affects the resulting predictions, and it may occasionally be useful to alter its default value (see Critical Parameters and Troubleshooting).

Examples of large-scale genomic annotation using geneid

geneid has been used in several genome annotation projects as the main gene prediction tool, or as a component of the ab initio gene prediction pipeline. For instance, geneid was the main "ab initio" gene prediction tool in the *Dictyostelium discoideum* genome project (Glockner et al., 2002). geneid has also been used in the large scale analysis of the genome sequence of *Tetraodon nigroviridis* (Jaillon et al., 2004) and *Paramecium tetraurelia* (Aury et al., 2006). Geneid is currently being used in the annotation of *O. dioica*, several fungi, the wine grape, tomato, pea aphid, and other genomes.

geneid predictions in human, mouse, and other species are served via DAS (Distributed Annotation System) through the UCSC genome browser (*http://genome.ucsc.edu*) and the Ensembl site (*http://www.ensembl.org*). They can also be found at the geneid Web site. geneid has also been used to scan the *Drosophila melanogaster* and the *Takifugu rubripes* genomes for putative selenoproteins (Castellano et al., 2001, 2004). See Suggestions for Further Analysis.

Critical Parameters and Troubleshooting

geneid is very easy to install and use, and, although is not bug-free, it should in general run without major problems. In some cases, however, geneid behavior may not be what the user is expecting. Mostly, in these cases, geneid will predict valid gene structures, but users will be unhappy with them. Unfortunately, it could also be that geneid does not produce results at all, or that it crashes while running. This section analyzes the most common causes of unsatisfactory geneid behavior and points to solutions, whenever possible.

geneid runs correctly and produces a valid gene prediction, but the user strongly suspect that the prediction is incorrect

For sequences other than short ones encoding single genes, only in a few percent of the cases will geneid prediction be completely correct. In most cases, the geneid prediction will nearly reproduce (at least one of) the exonic structures of the genes encoded in the input DNA sequence. A number of actual exons may be missed (maybe more than when using other gene-prediction programs), and some false exons or genes may additionally be predicted (in comparison to other gene-prediction programs, likely less). In some cases, the prediction will certainly be disastrous. There are a number of things the user can do to modify the default gene predictions. If the coordinates of some of the coding exons in the DNA sequence are known, the user can pass them to geneid via the `-R` option (see Basic Protocol 3).

If the user suspects that whole exons or genes have been missed, one can modify some of the values of the parameter file to attempt to recover them. There are two reasons why exons or genes may have been completely missed by geneid. Either (1) geneid does not consider them as candidate exons, or (2) it does predict them as candidate exons, but they are not included in the final gene prediction. It is easy to check which is the case by using the `-x` option, which outputs the complete list of candidate exons predicted by geneid. In the second case, the user can increase the value of the Exon Weight (EW) parameter in the parameter file (see Background Information). By default, this number is negative for most species. The higher the value, the higher the number of exons included in the final gene prediction. If the missing exons have not been included in the list of candidate exons, then decrease the cutoff values of exons and sites, and probably still increase the value of EW.

If there is biochemical or other evidence suggesting that the sequence encodes only a single gene, the authors suggest that you use a gene model that also reflects a single-exon gene.

geneid runs correctly, but stops with a warning before producing any prediction

The following error message will appear:

```
Too many predicted sites:
Change RSITES parameter
```

or a similar message concerning exon types. In order to minimize memory usage, geneid makes a guess on the maximum number of sites and exons that will be predicted in a given sequence fragment. While for most sequences, the guess is correct, in some (particularly anomalous) genomic sequences these numbers are much higher than that guessed. The user will need to change the parameters that control how these numbers are guessed. These parameters are assigned default values in the geneid header file, which the user will find at `include/geneid.h` within the geneid distribution. Decrease these values in the header file and recompile geneid (see geneid documentation for details). For instance, RSITES is 5 by default, so if the message above appears, change it to 2, then recompile geneid and run it again. Users must note that by decreasing these numbers the amount of memory required by geneid may increase substantially.

geneid produces inconsistent results or crashes after starting or while running

In some exceptional cases, geneid produces a prediction with inconsistent exon coordinates or crashes without a warning. The authors believe they have reduced these cases to a minimum, which were mostly related to memory management problems. If the user encounters these problems, please report them to the authors at *geneid@imim.es*.

Suggestions for Further Analysis

The authors are investigating a number of extensions to geneid, which are not discussed above:

1. *Incorporating homology information into the gene predictions*. For instance, such information can be obtained after the comparison of the query sequence against a database of known amino acid sequences using BLASTX (Altschul et al., 1990; *UNIT 3.4*) or FASTA (Pearson, 1990). Processed database search results can already be passed to geneid via `-S` option. The authors have chosen here not to discuss this option because the use of homology information requires fine tuning of some of the geneid parameters (tuning that the authors have not performed yet). Still, the option `-S` can be of utility. For instance, in Basic Protocol 3, when passing to geneid the coordinates of EST fragments via the `-R` option, these are processed as corresponding exactly to coding exons. Often, however, the exact coordinates of an exon are not known (for instance when matching similar, but not identical ESTs, or when the EST expands into the UTR). In such a case, the coordinates of the region in which the exons are suspected can be given to geneid via the `-S` option. geneid, then, will rescore all candidate exons overlapping the region. The resulting exon score will be a function of the original exon score, the score of the region, and the degree of overlap between the region and the exon. If the score given to the region is high enough, geneid exons overlapping the region will likely be included in the final gene prediction.

2. *Comparative gene prediction*. The authors have developed the program SGP2 (Parra et al., 2003) which combines TBLASTX (Altschul et al., 1990; *UNIT 3.4*) and geneid to use information from sequence similarity between genomes of two different species in gene predictions (for a review on comparative gene prediction, see Brent and Guigó, 2004). The SGP2 tool has been a component of the comparative gene prediction pipelines to annotate genes simultaneously in the human, mouse, rat, and chicken genomes (Mouse Genome Sequencing Consortium, 2002; Rat Genome Sequencing Project Consortium, 2004; International Chicken Genome Sequencing Consortium, 2004).

3. *Prediction of selenoproteins*. In selenoproteins, incorporation of the amino acid selenocysteine is specified by the UGA codon, usually a stop signal. The alternative decoding of UGA is conferred by an mRNA structure, the SECIS element, located in the 3′-untranslated region of the selenoprotein mRNA. Because of the nonstandard use of the UGA codon, current computational gene prediction methods are unable to identify selenoproteins in the sequence of the eukaryotic genomes. The authors have developed a version of geneid which is able to predict genes with exons containing TGA stop codons in frame. Through the option `-R`, SECIS predictions obtained by some other prediction program such as PATSCAN (*http://www-unix.mcs.anl.gov/compbio/PatScan/HTML/*) can be passed into this version of geneid,

which then predicts genes with in-frame TGA codons, only when an appropriate SECIS element has been predicted at the appropriate location. A prototype of this tool has been used to scan for potential selenoproteins in *Drosophila melanogaster* and *Takifugu rubripes* See Castellano et al. (2001, 2004) for further details.

Literature Cited

Abril, J.F. and Guigó, R. 2000. gff2ps: Visualizing genomic annotations. *Bioinformatics* 16:743-744.

Altschul, S.F., Gish, W., Miller, W., Myers, E.W., and Lipman, D.J. 1990. Basic local alignment search tool. *J. Mol. Biol.* 215:403-410.

Aury, J.M., Jaillon, O., Duret, L., Noel, B., Jubin, C., Porcel, B.M., Segurens, B., Daubin, V., Anthouard, V., Aiach, N., Arnaiz, O., Billaut, A., Beisson, J., Blanc, I., Bouhouche, K., Camara, F., Duharcourt, S., Guigó, R., Gogendeau, D., Katinka, M., Keller, A.M., Kissmehl, R., Klotz, C., Koll, F., Le Mouel, A., Lepere, G., Malinsky, S., Nowacki, M., Nowak, J.K., Plattner, H., Poulain, J., Ruiz, F., Serrano, V., Zagulski, M., Dessen, P., Betermier, M., Weissenbach, J., Scarpelli, C., Schachter, V., Sperling, L., Meyer, E., Cohen, J., and Wincker, P. 2006. Global trends of whole-genome duplications revealed by the ciliate *Paramecium tetraurelia*. *Nature* 444:171-178.

Birney, E. and Durbin, R. 2000. Using GeneWise in the *Drosophila* annotation experiment. *Genome Res.* 10:547-548.

Brent, M.R. and Guigó, R. 2004. Recent advances in gene structure prediction. *Curr. Opin. Struct. Biol.* 14:264-272.

Castellano, S., Novoselov, S.V., Kryukov, G.V., Lescure, A., Blanco, E., Krol, A., Gladyshev, V.N., and Guigó, R. 2004. Reconsidering the evolution of eukaryotic selenoproteins: A novel nonmammalian family with scattered phylogenetic distribution. *EMBO Rep.* 5:71-77.

Castellano, S., Morozova, N., Morey, M., Berry, M.J., Serras, F., Corominas, M., and Guigó, R. 2001. In silico identification of novel selenoproteins in the *Drosophila melanogaster* genome. *EMBO Reports* 2:697-702.

Fagioli, M., Alcalay, M., Pandolfi, P.P., Venturini, L., Mencarelli, A., Simeone, A., Acampora, D., Grignani, F., and Pelicci, P.G. 1992. Alternative splicing of PML transcripts predicts coexpression of several carboxy-terminally different protein isoforms. *Oncogene.* 7:1083-1091.

Glockner, G., Eichinger, L., Szafranski, K., Pachebat, J.A., Bankier, A.T., Dear, P.H., Lehmann, R., Baumgart, C., Parra, G., Abril, J.F., Guigó, R., Kumpf, K., Tunggal, B., Cox, E., Quail, M.A., Platzer, M., Rosenthal, A., Noegel, A.A.; Dictyostelium Genome Sequencing Consortium. 2002. Sequence and analysis of chromosome 2 of *Dictyostelium discoideum*. *Nature* 418:79-85.

Guigó, R. 1998. Assembling genes from predicted exons in linear time with dynamic programming. *J. Comp. Biol.* 5:681-702.

Guigó, R., Knudsen, S., Drake, N., and Smith, T. 1992. Prediction of gene structure. *J. Mol. Biol.* 226:141-157.

Guigó, R., Flicek, P., Abril, J.F., Reymond, A., Lagarde, J., Denoeud, F., Antonarakis, S., Ashburner, M., Bajic, V.B., Birney, E., Castelo, R., Eyras, E., Ucla, C., Gingeras, T.R., Harrow, J., Hubbard, T., Lewis, S.E., and Reese, M.G. 2006. EGASP: The human ENCODE Genome Annotation Assessment Project. *Genome Biol.* 7:S2.1-S3.31.

Hinrichs, A.S., Karolchik, D., Baertsch, R., Barber, G.P., Bejerano, G., Clawson, H., Diekhans, M., Furey, T.S., Harte, R.A., Hsu, F., Hillman-Jackson, J., Kuhn, R.M., Pedersen, J.S., Pohl, A., Raney, B.J., Rosenbloom, K.R., Siepel, A., Smith, K.E., Sugnet, C.W., Sultan-Qurraie, A., Thomas, D.J., Trumbower, H., Weber, R.J., Weirauch, M., Zweig, A.S., Haussler, D., and Kent, W.J. 2006. The UCSC Genome Browser Database: Update 2006. *Nucl. Acids Res.* 34:D590-D598.

International Chicken Genome Sequencing Consortium. 2004. Sequence and comparative analysis of the chicken genome provide unique perspectives on vertebrate evolution. *Nature* 432:695-716.

Jaillon, O., Aury, J.M., Brunet, F., Petit, J.L., Stange-Thomann, N., Mauceli, E., Bouneau, L., Fischer, C., Ozouf-Costaz, C., Bernot, A., Nicaud, S., Jaffe, D., Fisher, S., Lutfalla, G., Dossat, C., Segurens, B., Dasilva, C., Salanoubat, M., Levy, M., Boudet, N., Castellano, S., Anthouard, V., Jubin, C., Castelli, V., Katinka, M., Vacherie, B., Biemont, C., Skalli, Z., Cattolico, L., Poulain, J., De Berardinis, V., Cruaud, C., Duprat, S., Brottier, P., Coutanceau, J.P., Gouzy, J., Parra, G., Lardier, G., Chapple, C., McKernan, K.J., McEwan, P., Bosak, S., Kellis, M., Volff, J.N., Guigó, R., Zody, M.C., Mesirov, J., Lindblad-Toh, K., Birren, B., Nusbaum, C., Kahn, D., Robinson-Rechavi, M., Laudet, V., Schachter, V., Quetier, F., Saurin, W., Scarpelli, C., Wincker, P., Lander, E.S., Weissenbach, J., and Roest Crollius, H. 2004. Genome duplication in the teleost fish Tetraodon nigroviridis reveals the early vertebrate protokaryotype. *Nature* 431:916-917.

Lewis, S.E., Searle, S.M.J., Harris, N., Gibson, M., Iyer, V., Ricter, J., Wiel, C., Bayraktaroglu, L., Birney, E., Crosby, M.A., Kaminker, J.S., Matthews, B., Prochnik, S.E., Smith, C.D., Tupy, J.L., Rubin, G.M., Misra, S., Mungall, C.J., and Clamp, M.E. 2002. Apollo: A sequence annotation editor. *Genome Biology* 3:research0082.

Mott, R. 1997. EST_GENOME: A program to align spliced DNA sequences to unspliced genomic DNA. *Comp. Appl. Biosci.* 13:477-478.

Mouse Genome Sequencing Consortium. 2002. Initial sequencing and comparative analysis of the mouse genome. *Nature* 420:520-562.

Parra, G., Blanco, E., and Guigó, R. 2000. geneid in *Drosophila*. *Genome Res.* 10:511-515.

Parra, G., Agarwal, P., Óbril, J.F., Wiehe, T., Fickett, J.W., and Guigó, R. 2003. Comparative gene prediction in human and mouse. *Genome Res.* 13:108-117.

Pearson, W.R. 1990. Rapid and sensitive sequence comparison with FASTP and FASTA. *Methods Enzymol.* 183:63-98.

Rat Genome Sequencing Project Consortium. 2004. Genome sequence of the brown Norway rat yields insights into mammalian evolution. *Nature* 428:493-521.

Stormo, G.D. 2000. Gene-finding approaches for eukaryotes. *Genome Res.* 10:394-397.

Zhang, M.Q. 2002. Computational prediction of eukaryotic protein-coding genes. *Nat. Rev. Genet.* 3:698-709.

Key References

Guigó et al., 1992. See above.
Description of the first implementation of geneid.

Guigó et al., 2006. See above.
A community experiment to assess the state-of-the-art in one percent of the human genome sequence.

Parra et al., 2000. See above.
Description of geneid v 1.0 used in the Adh region of Drosophila melanogaster.

Internet Resources

http://genome.imim.es/software/geneid/index.html
This is the geneid Web page.

http://genome.imim.es/software/gfftools/GFF2PS.html
This is gff2ps Web page.

http://www.fruitfly.org/annot/apollo/
This is Apollo Web page (see UNIT 9.5)

http://genome.ucsc.edu/
This is UCSC genome browser (golden path; UNIT 1.4).

http://www.sanger.ac.uk/Software/formats/GFF/GFF_Spec.shtml
This is GFF format Web page.

http://www.w3.org/XML/
This is XML format Web page.

Contributed by Enrique Blanco, Genís Parra, and Roderic Guigó
Centre de Regulació Genòmica
Institut Municipal d'Investigació Mèdica
Universitat Pompeu Fabra
Barcelona, Spain

Using GlimmerM to Find Genes in Eukaryotic Genomes

GlimmerM is a gene finder originally developed for small eukaryotes, particularly for organisms with a relatively high gene density (Salzberg et al., 1999). The original system was designed to find genes in *Plasmodium falciparum*, the malaria parasite (Gardner et al., 1998). With the demands of many recent genome sequencing projects, each calling for its own gene finder, the system has been trained for many additional organisms, including *Arabidopsis thaliana*, *Oryza sativa* (Yuan et al., 2001), *Theileria parva*, and *Aspergillus fumigatus*. It performs well on all of these, even those with relatively low gene density, and on closely related organisms. A special package included with the latest release of GlimmerM re-trains the system using data provided by the user, thereby making the gene finder applicable to virtually any organism, limited only by the availability of training data.

Information on how to obtain the Unix version of GlimmerM software is presented in the Basic Protocol. This section also describes the usage of the system to predict gene models in genomic DNA sequences. The Support Protocol presents the steps required by the training procedure of GlimmerM. Early versions of GlimmerM required some human intervention in the training protocol; in particular, a programmer or biologist was required to choose thresholds for the false-negative and false-positive rates for the splice site recognition routines. Fortunately, the automatic training procedure in the current version obviates this requirement. To allow greater flexibility in tuning the system, the training procedure permits the user to consult the false-positive and false-negative rates determined from the training data and to adjust the corresponding system parameters. The Support Protocol gives the user the necessary knowledge for changing these default thresholds and other parameters of the gene finder. Adjustment of these parameters frequently yields better gene predictions because of the wide variation in DNA sequence characteristics for different organisms. An Alternate Protocol briefly describes running GlimmerM from the TIGR Web site. However, individuals who choose to run GlimmerM over the internet do not have the option of training the system.

RUNNING GlimmerM LOCALLY TO IDENTIFY GENES

BASIC PROTOCOL

The most powerful and flexible way of using GlimmerM is to install and run the Unix-based software on a local system. This gives the user more organism-specific versions of GlimmerM (these are included with the software), and the power to train the system for any organism of choice, provided that one can collect a representative training set (see Support Protocol). Another advantage of having GlimmerM locally installed is that the parameters of the system can be customized to reflect the user's expertise about the organism, e.g., by changing default parameters of the program such as the minimum gene length or the prediction overlap allowed.

Necessary Resources

Hardware

A Unix workstation. GlimmerM has been successfully compiled for Linux, Digital Unix, and SunOS, and it should be easy to compile on any platform supporting ANSI C and C++.

Contributed by Mihaela Pertea and Steven L. Salzberg
Current Protocols in Bioinformatics (2003) 4.4.1-4.4.20
Copyright © 2003 by John Wiley & Sons, Inc.

Software

Currently, there are two packages available: GlimmerM 1.2 and GlimmerM 2.0. In the GlimmerM 1.2 package, the code of the gene finder is trained and customized specifically for each organism. GlimmerM 2.0 is upgraded to contain the automatic training procedure (see Support Protocol) and a generally applicable gene finding algorithm. GlimmerM 2.0 contains all of the organism-specific versions found in version 1.2; however, the performance of these versions is slightly different due to changes in parameter settings when building the later system. The example below uses GlimmerM 2.0, but the basic procedure for running versions 1.2 and 2.0 is the same. A truly determined user who is studying all of the organisms included in the 1.2 release might want to get both versions in order to compare the results.

The original GlimmerM system, designed specifically for P. falciparum, uses a slightly different algorithm than subsequent versions of the GlimmerM program, as explained in the Guidelines for Understanding Results and Commentary sections of this unit. Because this initial algorithm had its own advantages, the authors chose to keep it and include it in a separate directory as part of the software release. After downloading the GlimmerM software, one can find this initial gene finder, including source code, binaries, and the latest malaria training set, in a separate subdirectory called `Malaria`*. The source code for the current version of the gene finder can be found in the* `sources` *subdirectory and the training procedure is included in the* `train` *subdirectory. Each subdirectory contains a* `Readme` *file explaining how to locally compile the source code. Organism-specific versions of the system can be found in the* `trained_dir` *subdirectory.*

GlimmerM is available free of charge to researchers using it for non-commercial purposes. The system includes source code and a `Readme` *file describing how to compile and train the system. Pre-trained versions for a small number of organisms (Plasmodium falciparum, Arabidopsis thaliana, Oryza sativa, Theileria parva, and Aspergillus fumigatus) are included; that number continues to grow as more genomes are sequenced. In order to obtain the system, a representative of a nonprofit organization should fill out a license agreement available on the TIGR Web site, http://www.tigr.org, under Software. Interested commercial organizations should see the Web site for additional instructions. For nonprofit organizations, the system is made available almost immediately after submitting the license agreement.*

Files

A FASTA file (*APPENDIX 1B*) containing the sequence to be analyzed. (There is no maximum sequence length set by default in the program.)

The FASTA file used in the example below is available at the Current Protocols Web site (http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm).

Install software

1. Submit a license agreement and download, install, and compile the software. To install and compile the program type:

```
> tar xvfz GlimmerM.tar.gz
> cd GlimmerM/sources
> make
```

2. If necessary, train GlimmerM for a new organism (see Support Protocol).

Run GlimmerM to analyze DNA sequences for their coding potential

3. The program GlimmerM takes two inputs: a DNA sequence file in FASTA format (*APPENDIX 1B*) and a directory containing the training files for the program. If not specified, the training directory is assumed to be the current working directory. For

instance, if the user is running a pre-compiled version of GlimmerM located in the `bin` directory, the following command should be used:

```
glimmerm_<system> <FASTA file with the DNA sequence to
be analyzed>
```

or

```
glimmerm_<system> <FASTA file> -d <directory of the
training files>
```

where `<system>` is `linux`, `alpha`, or `sun`.

The `-d` parameter specifies the directory containing the training files.

For the pre-trained versions of the system, this directory will be: `GlimmerM/trained_dir/[organism_name]`

For user-trained executables *(see Support Protocol)*, this directory will be: `TrainGlimmM[date][time]`

Other optional parameters that can be given to the program are shown in Table 4.4.1 The annotations below discuss commonly used parameters. The remaining parameters are discussed in the Critical Parameters section below.

The minimum gene length can be specified with the `-g` *option. This value is the length of the smallest fragment considered to be a possible gene and is measured from the first base of the start codon to the last base before the stop codon. The* `-o` *and* `-p` *parameters refer only to the special version of GlimmerM trained for malaria (see the Files section) and specify the amount by which two coding regions are allowed to overlap to be considered different gene models (the default overlap length and percent are 30 bp and 10%, respectively).*

To determine if a putative model is likely to be a gene, GlimmerM scores the coding region of that model in each of the six possible reading frames. If the putative model's coding sequence in the correct reading frame scores above the score set by the `-t` *option (by default* `t=90`*) then that model is predicted to be a gene. If the* `+r` *option is added when running GlimmerM, then the score of the putative coding region in the correct reading frame is also compared to the score generated by a "random" model, which is a simple Markov chain that uses independent probabilities for each base. See, for example, Salzberg et al. (1999) for a description of how to use Markov chains for biological sequence analysis. The* `+r` *option is active by default, but it can be disabled by adding* `-r` *to the command line.*

By default, GlimmerM uses a maximal local filtering for the splice sites, with a window length read from the `config_file`*. This is equivalent to using the* `+f` *option when running GlimmerM. Because the filter may increase the number of false negatives, the* `-f` *option should be used when no filtering is desired. The splice-site thresholds that the program reads from* `config_file` *can also be changed with the* `-5` *and* `-3` *parameters, in this way overriding the initial threshold values given in the* `config_file`*. If enough data is available, GlimmerM will train a module to reduce the false-positive rates of the translational start recognition. If the user does not wish to use this module, the* `-s` *option should be used. When the* `-s` *parameter is not specified, the* `+s` *option is enabled if possible.*

4. Examine the results (see Guidelines for Understanding Results). The results of GlimmerM are printed on the screen but the user can redirect the output of the system by using a ">" sign followed by a filename.

Table 4.4.1 Optional Parameters to Use When Running GlimmerM[a]

Command	Argument	Default values	
		Version 1.2	Version 2.0
`-d`	*dir* Set the directory of the training files to `dir`	./	./
`-g`	*n* Set minimum gene length to `n`	175	60
`-o`[b]	*n* Set minimum overlap length to `n` nucleotides. Overlaps shorter than this are ignored.	30	0
`-p`[b]	*n* Set minimium overlap percentage to `n%`. Overlaps shorter than this percentage of both strings are ignored.	10	0
`-t`	*n* Set the threshold score above which a DNA region is called a gene to `n`. If the in-frame score is greater or equal to `n` (an integer between 0 and 100), then the region is given a number and considered a potential gene.	99	90
`-r`	Do not use independent probability score column	No	No
`+r`	Use independent probability score column	Yes	Yes
`-f`	Do not use maximal local filtering of the splice sites	n/a	No
`+f`	Use maximal local filtering of the splice sites	n/a	Yes
`-s`	Do not use the translation start site model	n/a	No
`+s`	Use the start site model	Yes	Yes
`-5`	*t* Use threshold *t* for the acceptor sites	n/a	Set in `config_file`
`-3`	*t* Use threshold *t* for the donor sites	n/a	Set in `config_file`

[a]Further discussion of these parameters can be found in the Critical Parameters section.

[b]The `-o` and `-p` options are only available for the *P. falciparum* version of GlimmerM.

SUPPORT PROTOCOL

TRAINING GlimmerM FOR A SPECIFIC ORGANISM

First of all, a careful, thorough collection of a good training set is a critical first step in the training of any gene finder. The quality of the data used for training is directly proportional to the accuracy of the resulting gene finder. As with any species-specific gene finder, GlimmerM needs to learn about the properties of the genes in an organism before it can find more genes. A good training set should contain as many complete coding sequences as possible from the organism for which a gene finder is needed. It is difficult to specify precisely how many genes are sufficient to form an adequate training set, because this number is influenced by several factors, such as the length of the ORFs and the number of confirmed splice sites that these genes contain. Estimating the parameters of a complex model involving Markov chains like the one used by the authors' splice-detection module (see Background Information) is not an easy task. As Burge (1997) shows, at least 700 splice-site sequences will give a tolerable range of error (between 10% and 20%) in the estimation of the first order Markov transition probabilities.

By surveying the public databases, one can obtain all previously discovered genes for the target organism, and if possible, these should be limited to those that are validated by laboratory evidence (as opposed to computational predictions of genes). These genes will

form an adequate training data set if a sufficient number is found. Unfortunately, this is rarely the case for organisms targeted for whole-genome sequencing, therefore, other methods should be used in order to construct a reliable data set. In the authors' experience, an effective strategy for constructing a training set is to wait until a genome project has generated several hundred thousand base pairs of data. From this data, one can easily extract all of the long open reading frames (ORFs), i.e., stretches of DNA sequence without a stop codon. "Long" ORFs may be ≥500 bp, depending on the GC-content of the genome. These long ORFs may then be searched against a non-redundant protein sequence database using BLAST (Altschul et al., 1990; *UNITS 3.3 & 3.4*), and any ORFs that have a significant hit may be safely assumed to be derived from real genes. A step-by-step procedure to train GlimmerM follows.

Necessary Resources

Hardware

A Unix workstation. GlimmerM has been successfully compiled for Linux, Digital Unix, and SunOS, and it should be easy to compile on any platform supporting ANSI C and C++.

Software

GlimmerM 2.0 is an upgraded version that contains the automatic training procedure and a generally applicable gene finding algorithm. GlimmerM 2.0 contains all of the organism-specific versions found in version 1.2; however, the performance of these versions is slightly different due to changes in parameter settings when building the later system.

GlimmerM is available free of charge to researchers using it for non-commercial purposes. The system includes source code and a `readme` *file describing how to compile and train the system. In order to obtain the system, a representative of a nonprofit organization should fill out a license agreement available on the TIGR Web site, http://www.tigr.org, under Software. Interested commercial organizations should see the Web site for additional instructions. For nonprofit organizations, the system is made available almost immediately after submitting the license agreement.*

Files

Format the training data in two files:

a. A single FASTA file (*APPENDIX 1B*) containing all the DNA sequences for the training data, e.g.:

```
>Seq1 DNA sequence containing one or more genes
AGTCGTCGCTAGCTAGCTAGCATCGAGTCTTTTCGATCGAGGACTAGA
CTAGCTAGCTAGCATAGCATACGAGCATATCGGTCATGAGACTGATTGGGGTGTGTGC
TAAACTGTGT

>Seq2 another DNA sequence containing more genes
TTTAGCTAGCTAGCATAGCATACGAGCATATCGGTAGACTGATTGGGTTTATGCGTTA
```

b. A file specifying the locations of the known genes by the coordinates of the coding portions of those genes in each sequence in the FASTA file. For each coding exon, its 5′ and 3′ ends should be listed in order from start to stop. Thus, genes on the complementary strand will have these coordinates listed in decreasing order. Non-coding exons and non-coding portions of exons should not be listed; for example, if an exon spans positions 200-300 of a sequence and the start codon occurs at position 250, then the coordinate file should just list "250 300." A blank line must separate different genes. The format of this file is given in the example below:

```
Seq1 1 15
Seq1 20 34

Seq1 50 48
Seq1 45 36

Seq2 17 40
```

In this example, Seq1 has two genes: the first one is on the direct strand, and its coding sequence covers positions 1-15 and 20-34, for a total of 30 nucleotides. The second gene in Seq1 is on the complementary strand, while Seq2 has only a single intronless gene on the forward strand.

GlimmerM can also use incomplete gene sequences in training, provided that the coordinates given in this file start in-frame. For example, suppose that the first gene on Seq1 in the above example extends off the sequence in the 5′ direction, with its unknown start codon somewhere upstream. If the correct reading frame starts in position 2, then its exon coordinates should be specified in the training file as:

```
Seq1 2 15
Seq1 20 34
```

The FASTA file and coordinate file used in the example below are available at the Current Protocols Web site (http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm).

1. Download and install GlimmerM (see Basic Protocol).
2. Change to the `train` subdirectory. Compile the training module by running `make` in the `train` subdirectory of the package:

   ```
   GlimmerM/> cd train
   GlimmerM/train> make
   ```

3. From a Unix console or shell window, train GlimmerM with the following command:

   ```
   trainGlimmerM <mfasta_file> <exon_file>
   [optional_parameters]
   ```

   `<mfasta_file>` and `<exon_file>` contain the names of the FASTA file and the file containing the exon coordinates of the known genes, respectively. A concrete example of running `trainGlimmerM` for malaria data is presented in Figure 4.4.1.

   One of the main steps of GlimmerM's gene finding algorithm is determining potential splice sites in the DNA sequences provided as input. Splice-site sequences that contain the consensus GT or AG dinucleotides and score above a fixed threshold are retained as potential donor or acceptor sites. These are filtered further by keeping only those sequences whose score was maximal within a fixed DNA window (Pertea et al., 2001). The default length of this window is 60 bp, but it can be changed by using two optional parameters with the `trainGlimmerM` *procedure:*

   `-a` *[filter value]*

   where [filter value] is an integer specifying the window length for filtering locally maximal acceptor sites (default = 60)

   `-d` *[filter value]*

   where [filter value] is an integer specifying the window length for filtering locally maximal donor sites (default = 60)

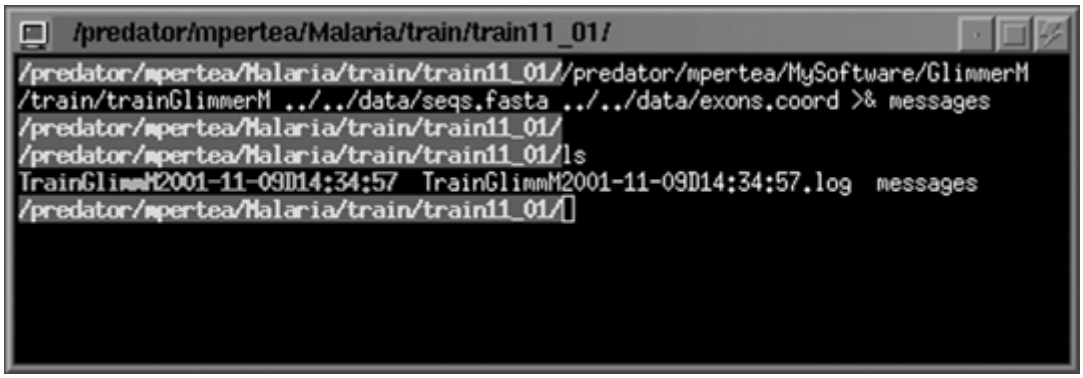


Figure 4.4.1 Training GlimmerM for a malaria data set, where the DNA sequences for training are in the file `seqs.fasta` and the exon coordinates are in the file `exons.coord` (shown on the first and second lines). By adding the text `>& messages` at the end of the trainGlimmerM command, the authors created a file called `messages`, which captured any potential messages of the training program that were normally printed on the screen. The computer prompt shown is `/predator/mpertea/Malaria/train/train11_01`. The third line shows the execution of the `ls` command (*APPENDIX 1C*), which lists the contents of the directory (fourth line). The directory now contains the subdirectory `TrainGlimmM2001-11-09D14:34:57`, and the two files `TrainGlimmM2001-11-09D14:34:57.log` and `messages`.

If insufficient data is available for training the splice sites, the training procedure will be unsuccessful and exit with a warning message. The system determines dynamically whether the data is sufficient by estimating error rates on both donor and acceptor sites. If it fails, then the user should collect more known genes with introns and then try the training procedure again. If insufficient data is available to train the internal model of translational start sites, the training procedure will succeed but GlimmerM will consider any ATG a potential start site. A flag on line 14 of `config_file` *(see step 5 below) will indicate if there was enough data to train the start sites (this flag is equal to 0 in the case of insufficient data).*

4. Change to the newly created `TrainGlimmM[date][time]` subdirectory. View the `config_file` (Fig. 4.4.2) and the `TrainGlimmM[date][time].log` file (Fig. 4.4.3) by using a text editor (see *APPENDIX 1C*).

   The `trainGlimmerM` *program creates a log file and a subdirectory under the directory where the user ran the training procedure. The log file called* `TrainGlimmM[date][time].log` *(Fig. 4.4.3) can be consulted to find the default values used for some of the parameters of GlimmerM. This subdirectory is called* `TrainGlimmM[date][time]`*, where* `[date]` *and* `[time]` *specify the date and time when the subdirectory was created.* `TrainGlimmM[date][time]` *contains the training parameters needed by GlimmerM to run. The subdirectory also contains a configuration file called* `config_file` *(Fig. 4.4.2) that specifies the parameters in Table 4.4.2.*

5. If necessary, modify the parameters in the `config_file` obtained from the training routines (see Critical Parameters and Troubleshooting). This step is optional, but the authors describe it here because manual tuning sometimes can improve the accuracy of the gene predictions.

   The flags on lines 6 and 7 in the `config_file` *(see Table 4.4.2 are just internally used by the system. They signal if decision trees were created from the available data or not. GlimmerM will use decision trees in computing the splice-site scores only if these flags were created by the training procedure (see Background Information for a brief description of how the splice sites are determined). A decision tree is a supervised learning method that learns to classify objects from a set of examples. It takes the form of a tree structure of nodes and edges in which the root and the internal nodes test on one of the objects'*

Figure 4.4.2 Example of `config_file`. Refer to Table 4.4.2 for parameters.

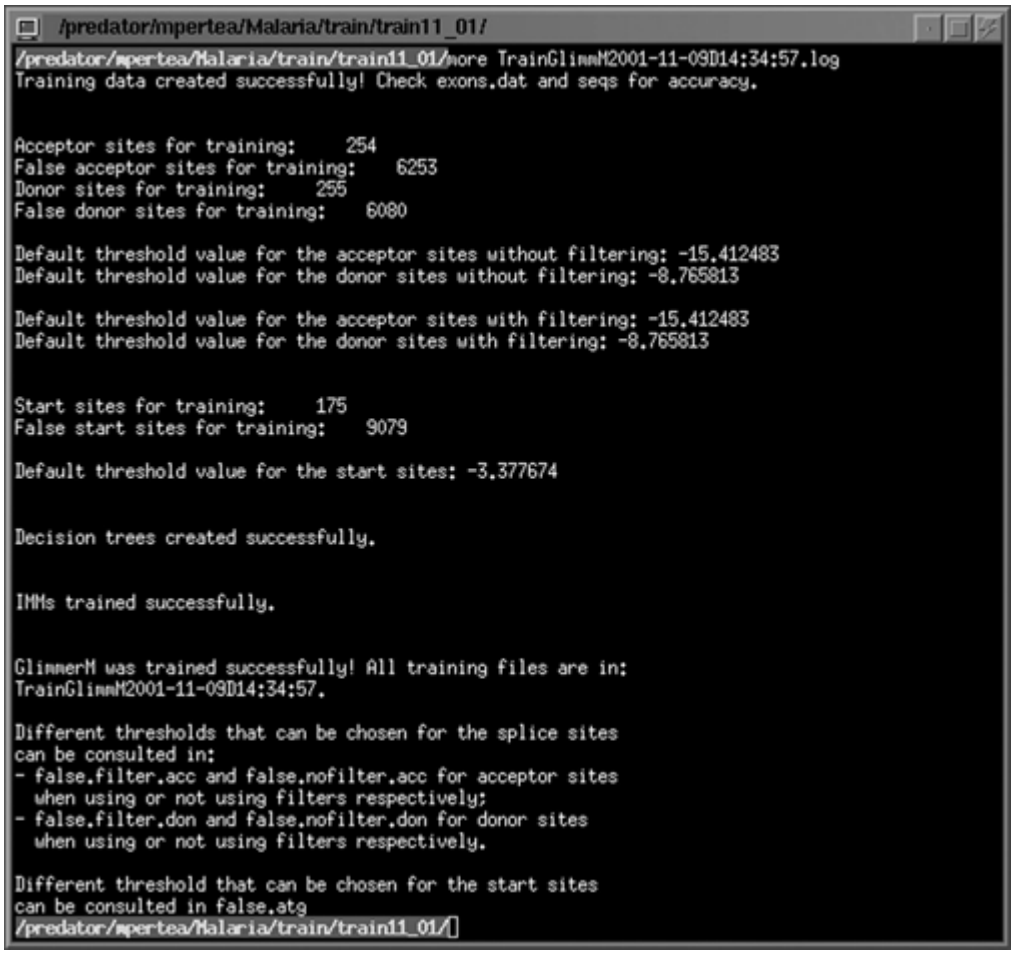

/predator/mpertea/Malaria/train/train11_01/
/predator/mpertea/Malaria/train/train11_01/more TrainGlimmM2001-11-09D14:34:57.log
Training data created successfully! Check exons.dat and seqs for accuracy.
Acceptor sites for training: 254
False acceptor sites for training: 6253
Donor sites for training: 255
False donor sites for training: 6080
Default threshold value for the acceptor sites without filtering: -15.412483
Default threshold value for the donor sites without filtering: -8.765813
Default threshold value for the acceptor sites with filtering: -15.412483
Default threshold value for the donor sites with filtering: -8.765813
Start sites for training: 175
False start sites for training: 9079
Default threshold value for the start sites: -3.377674
Decision trees created successfully.
IMMs trained successfully.
GlimmerM was trained successfully! All training files are in:
TrainGlimmM2001-11-09D14:34:57.
Different thresholds that can be chosen for the splice sites
can be consulted in:
- false.filter.acc and false.nofilter.acc for acceptor sites
when using or not using filters respectively;
- false.filter.don and false.nofilter.don for donor sites
when using or not using filters respectively.
Different threshold that can be chosen for the start sites
can be consulted in false.atg
/predator/mpertea/Malaria/train/train11_01/

Figure 4.4.3 An example log file generated by `trainGlimmerM`.

Table 4.4.2 Parameters for the Configuration File Called `config_file` (see Fig. 4.4.2)

Line no. in the `config_file`	Parameter
1	Minimum intron length
2	Maximum intron length
3	Minimum exon length
4	Maximum exon length
5	Maximum gene length
6	A flag indicating if decision trees were used for computing the acceptor site scores
7	A flag indicating if decision trees were used for computing the donor site scores
8	Acceptor site threshold
9	Donor site threshold
10	Length of the filter window for the acceptor sites
11	Length of the filter window for the donor sites
12	Acceptor site threshold when filtering is used
13	Donor site threshold when filtering is used
14	A flag indicating if start codon modeling was used
15	Start codon threshold

attributes. Each edge extending from an internal node of the tree represents one of the possible alternatives of courses of action available at that point. So, depending on the outcome of the test, different paths in the tree are followed down to the tree leaves that carry the class names into which the objects are classified.

6. Change all of the other parameters, with the exception of parameters 6, 7, and 14 shown in Table 4.4.2, by opening the `config_file` with a text editor (see *APPENDIX 1C*). Before modifying the values of the thresholds specified on lines 8, 9, 12, 13, and 15 of the `config_file`, consult the false-positive and false-negative rates from the following files: `false.nofilter.acc`, `false.nofilter.don`, `false.filter.acc`, `false.filter.don`, and `false.atg`, respectively. These threshold files can be found in the same directory as `config_file`, and all of them have the same format. Figures 4.4.4 and 4.4.5 present the first lines of the false-negative/false-positive rates for the acceptor and donor sites.

   From Figure 4.4.2, one can see that the default value of the thresholds for the acceptor and donor sites was set to −15.41 and −8.76, respectively. This corresponds to a 0% false-negative rate for the acceptor sites, and a 0.39% false-negative rate for the true donor sites. A user might not be satisfied that 6.0% of the GTs in the data will be called donor sites, in which case, one can set a higher threshold in order to have fewer false predictions. For instance, a threshold of 1.26 will introduce fewer false positives (only 2.0% of all GTs that are not donor sites in the data), but 10 (or ~4%) of the true donor sites will be missed. This threshold can be introduced in line 9 of the `config_file` *(see Table 4.4.2) to reflect the new rates. All threshold parameters from the* `config_file` *(lines 8, 9, 12, 13, and 15; see Table 4.4.2) can be changed in the same way by analyzing the corresponding threshold file.*

7. When filtering is used, the false-positive and false-negative rates are given only for a default length of the filter window (60 bp), thus any change in the length of either of the filter windows (i.e., parameter on line 10 or 11 of the `config_file`; Table

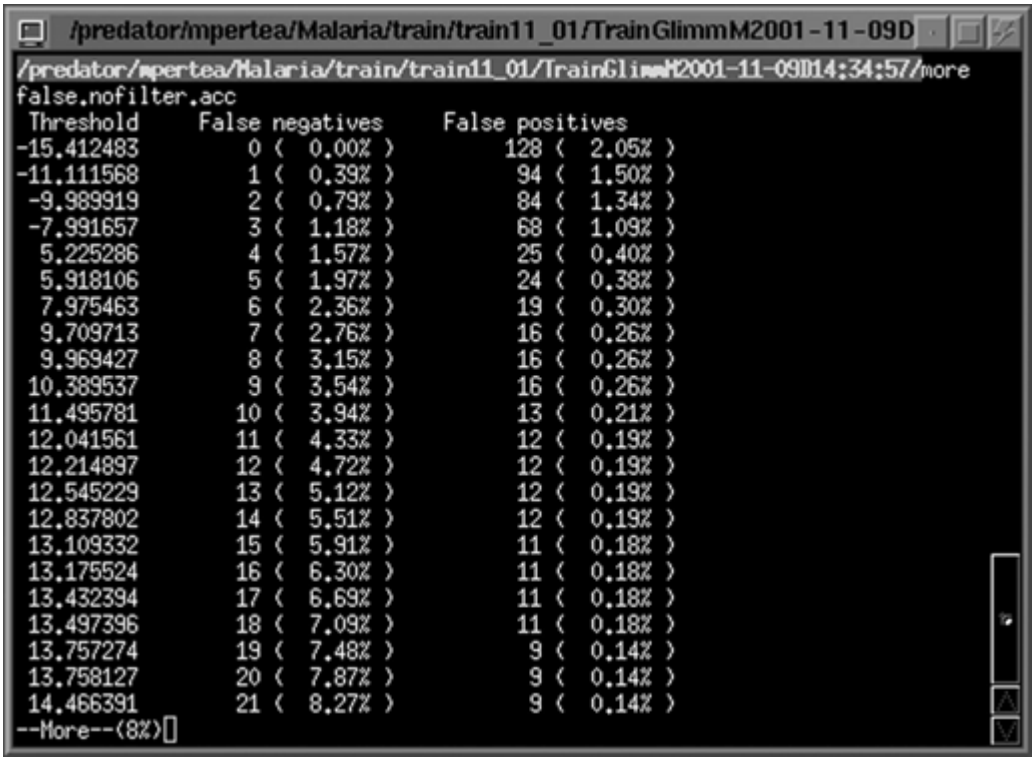


Figure 4.4.4 Example of `false.nofilter.acc` file.

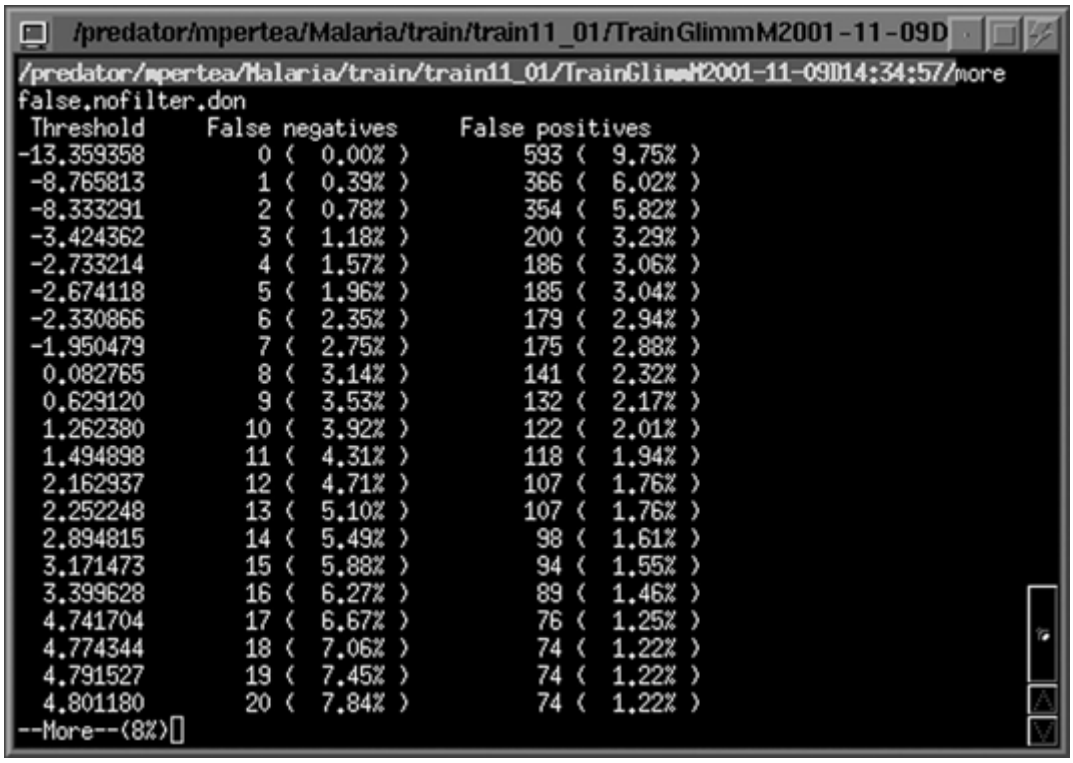


Figure 4.4.5 Example of `false.nofilter.don` file.

4.4.2) will cause a change in the value of the corresponding threshold (i.e., parameter on line 12 or 13 of the `config_file`; Table 4.4.2). Therefore, re-run the `train-GlimmerM` procedure using the `-a` and `-d` optional parameters after changing the length of the filter window for either donor or acceptor sites (see step 3).

RUNNING GlimmerM VIA THE WEB

ALTERNATE PROTOCOL

The GlimmerM system can be run directly on genomic sequences by using the Web interface at TIGR, located at *http://www.tigr.org/softlab/glimmerm*. The Web server provides genefinding using GlimmerM 1.2 for three organisms—*P. falciparum*, *A. thaliana*, and *O. sativa* (rice)—and others may be added in the future. This Web interface to GlimmerM should fulfill the needs of laboratories that do not have the facilities to install and run a Unix-based software system like GlimmerM, and of those laboratories that might be sequencing a single BAC or some other small region of a genome.

The authors' Web server allows anyone to submit sequences for analysis in chunks as large as 200,000 bp, by uploading a FASTA-formatted file (*APPENDIX 1B*) into the server. Sequences <30 kbp can be directly pasted into the browser. The user has the option of selecting which organism-specific version of the gene finder is desired, and also the option of whether to see the results on the screen or to have them sent by E-mail (see Figs. 4.4.6 and 4.4.7). The performance of GlimmerM will degrade for organisms other than those used for training the system; thus for anything other than organisms closely related to the three listed above, re-training is highly desirable as explained in the Support Protocol, however, the Web interface does not allow users to re-train GlimmerM.

GUIDELINES FOR UNDERSTANDING RESULTS

When annotating a genome, GlimmerM should be used as one of a suite of tools. Accurate gene identification depends on using every tool available, and the description in this unit should not be taken as an implication that GlimmerM alone can find all genes in a given genome. In order to produce reasonably accurate gene annotations, any comprehensive annotation effort needs several computational tools, e.g., searches using BLAST (*UNITS 3.3 & 3.4*) and/or PSI-BLAST against a non-redundant protein sequence database (Altschul et al., 1990, 1997); gapped alignments of DNA to protein and EST sequence databases (Florea et al., 1998); prediction of putative signal peptides, tools to detect frame-shift errors (e.g., Framesearch, *UNIT 3.2*); and graphical tools to allow annotators to view all the evidence concurrently. When no database matches or other computational evidence are found to support a GlimmerM prediction, then (just as for any other gene finder) further investigation is required to confirm these models.

At the request of the annotation team, the GlimmerM system trained for malaria was designed to produce multiple gene models for some genes. In all other current versions of GlimmerM, only the highest scoring model of a putative gene appears in the output. The discussion below explains how to interpret the output of GlimmerM, in the special case when overlapping gene models are predicted. Although many of GlimmerM's predictions are likely to be correct, it is undoubtedly the case that some are not. An estimation of the gene finder's accuracy is presented below as well.

Malaria Version

The output of the malaria-specific version (Fig. 4.4.8) is somewhat different and will be explained first, followed by an explanation that covers all other versions. The first few lines of output from the *Plasmodium* (malaria) version of GlimmerM specify the settings of various parameters in the program:

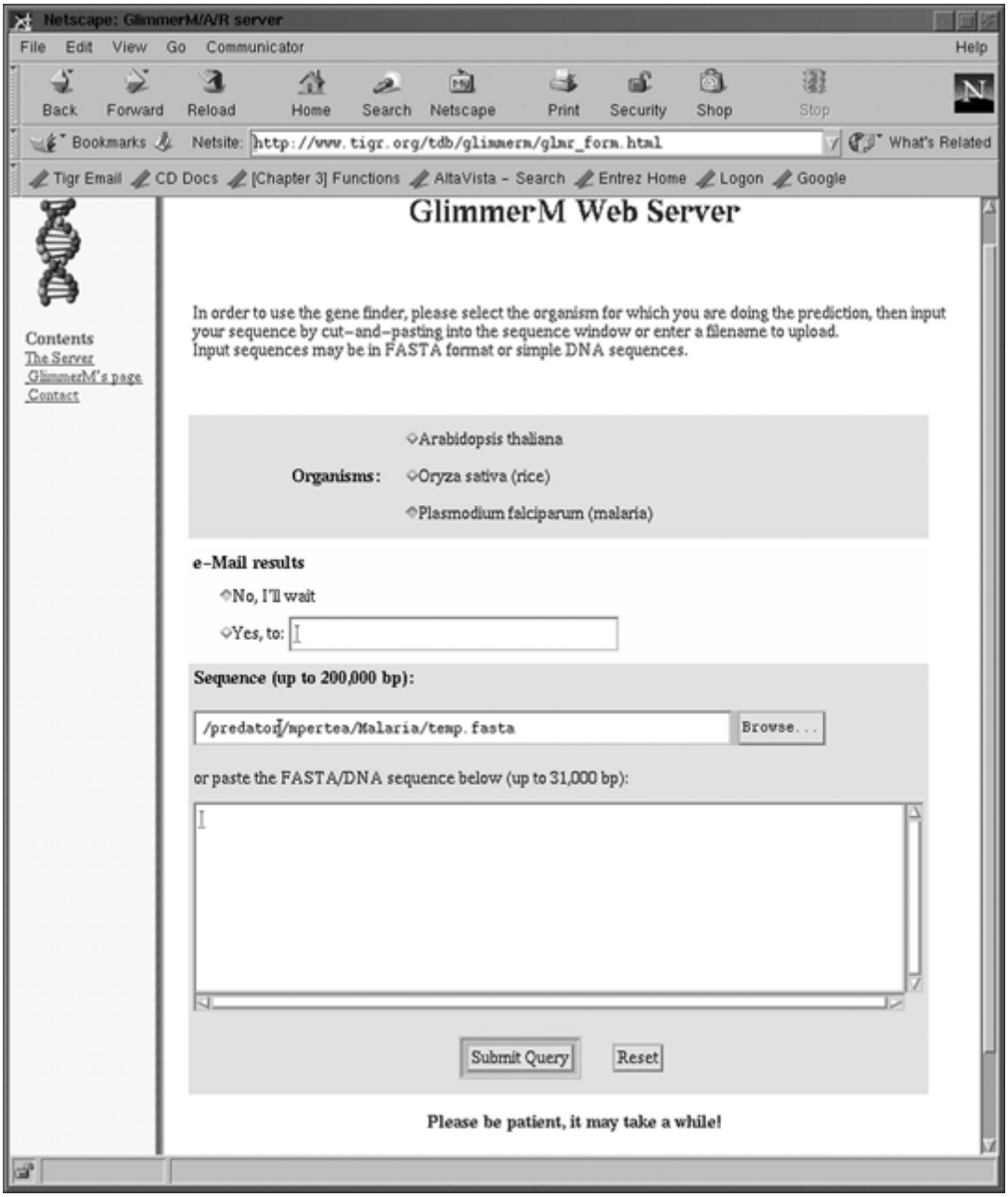

Netscape: GlimmerM/A/R server
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Netsite: http://www.tigr.org/tdb/glimmerm/glmr_form.html What's Related
Tigr Email CD Docs [Chapter 3] Functions AltaVista - Search Entrez Home Logon Google
GlimmerM Web Server
Contents
The Server
GlimmerM's page
Contact
In order to use the gene finder, please select the organism for which you are doing the prediction, then input your sequence by cut-and-pasting into the sequence window or enter a filename to upload.
Input sequences may be in FASTA format or simple DNA sequences.
Organisms:
Arabidopsis thaliana
Oryza sativa (rice)
Plasmodium falciparum (malaria)
e-Mail results
No, I'll wait
Yes, to:
Sequence (up to 200,000 bp):
/predator/mpertea/Malaria/temp.fasta
Browse...
or paste the FASTA/DNA sequence below (up to 31,000 bp):
Submit Query
Reset
Please be patient, it may take a while!

Figure 4.4.6 Example of using the GlimmerM Web server.

Minimum gene length is the length in nucleotides of the smallest fragment permitted to be called a gene.

Minimum overlap length is a lower bound on the number of bases overlap between two genes that is considered to be a problem. Overlaps shorter than this are ignored.

Minimum overlap percent is another lower bound on overlap. Overlaps shorter than this percentage of both genes are ignored.

Threshold score is the minimum in-frame score for a fragment to be considered a potential gene and it is computed as in Salzberg et al. (1998a).

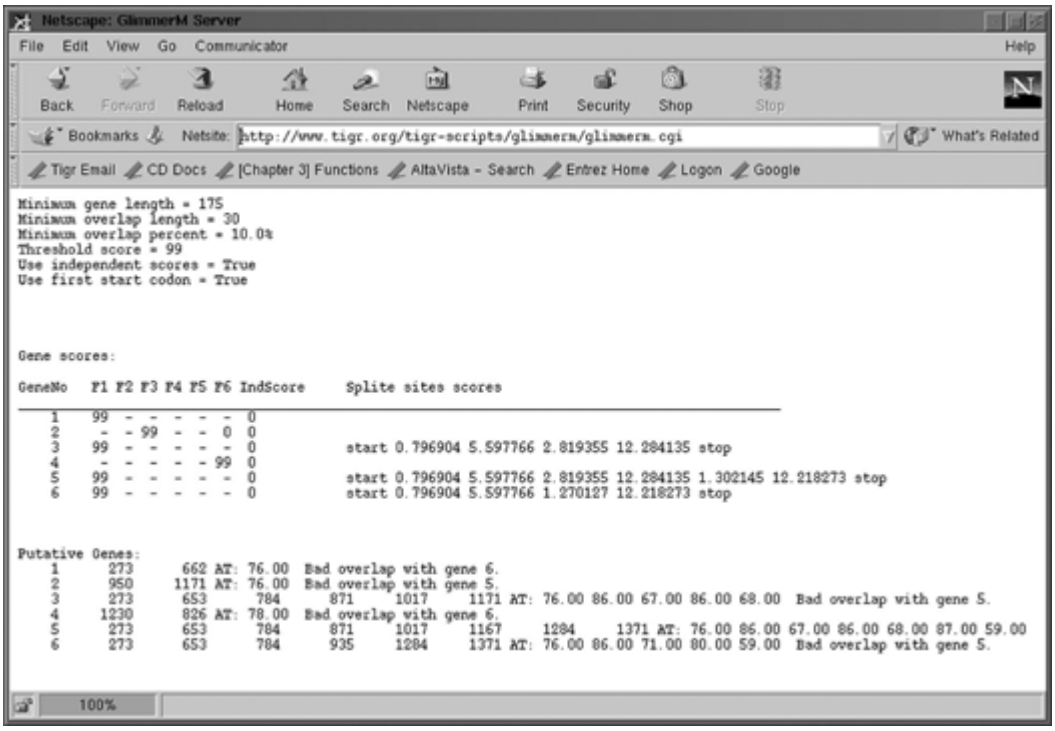

Netscape: GlimmerM Server
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Netsite: http://www.tigr.org/tigr-scripts/glimmerm/glimmerm.cgi What's Related
Tigr Email CD Docs [Chapter 3] Functions AltaVista - Search Entrez Home Logon Google
Minimum gene length = 175
Minimum overlap length = 30
Minimum overlap percent = 10.0%
Threshold score = 99
Use independent scores = True
Use first start codon = True
Gene scores:
GeneNo F1 F2 F3 F4 F5 F6 IndScore Splite sites scores
1 99 - - - - - 0
2 - - 99 - - - 0 0
3 99 - - - - - 0 start 0.796904 5.597766 2.819355 12.284135 stop
4 - - - - - - 99 0
5 99 - - - - - 0 start 0.796904 5.597766 2.819355 12.284135 1.302145 12.218273 stop
6 99 - - - - - 0 start 0.796904 5.597766 1.270127 12.218273 stop
Putative Genes:
1 273 662 AT: 76.00 Bad overlap with gene 6.
2 950 1171 AT: 76.00 Bad overlap with gene 5.
3 273 653 784 871 1017 1171 AT: 76.00 86.00 67.00 86.00 68.00 Bad overlap with gene 5.
4 1230 826 AT: 78.00 Bad overlap with gene 6.
5 273 653 784 871 1017 1167 1284 1371 AT: 76.00 86.00 67.00 86.00 68.00 87.00 59.00
6 273 653 784 935 1284 1371 AT: 76.00 86.00 71.00 80.00 59.00 Bad overlap with gene 5.
100%

Figure 4.4.7 Output of GlimmerM Web Server.

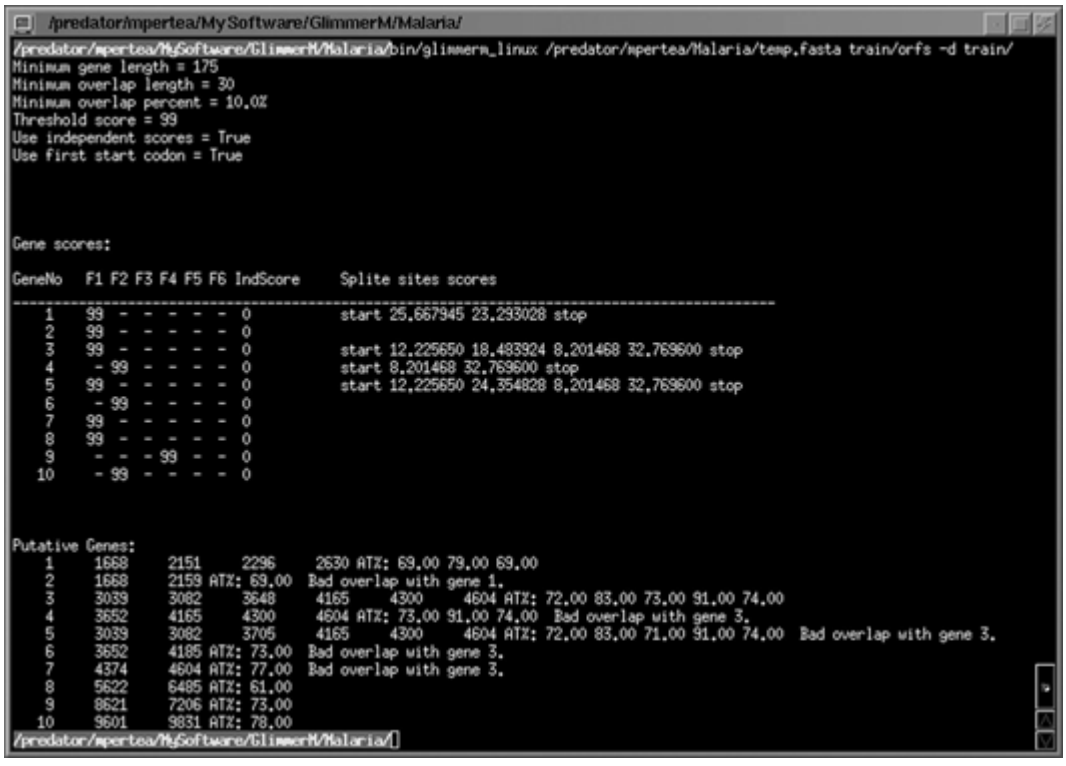

/predator/mpertea/MySoftware/GlimmerM/Malaria/
/predator/mpertea/MySoftware/GlimmerM/Malaria/bin/glimmerm_linux /predator/mpertea/Malaria/temp.fasta train/orfs -d train/
Minimum gene length = 175
Minimum overlap length = 30
Minimum overlap percent = 10.0%
Threshold score = 99
Use independent scores = True
Use first start codon = True
Gene scores:
GeneNo F1 F2 F3 F4 F5 F6 IndScore Splite sites scores
1 99 - - - - - 0 start 25.667945 23.293028 stop
2 99 - - - - - 0
3 99 - - - - - 0 start 12.225650 18.483924 8.201468 32.769600 stop
4 - 99 - - - - 0 start 8.201468 32.769600 stop
5 99 - - - - - 0 start 12.225650 24.354828 8.201468 32.769600 stop
6 - 99 - - - - 0
7 99 - - - - - 0
8 99 - - - - - 0
9 - - - 99 - - 0
10 - 99 - - - - 0
Putative Genes:
1 1668 2151 2296 2630 AT: 69.00 79.00 69.00
2 1668 2159 AT: 69.00 Bad overlap with gene 1.
3 3039 3082 3648 4165 4300 4604 AT: 72.00 83.00 73.00 91.00 74.00
4 3652 4165 4300 4604 AT: 73.00 91.00 74.00 Bad overlap with gene 3.
5 3039 3082 3705 4165 4300 4604 AT: 72.00 83.00 71.00 91.00 74.00 Bad overlap with gene 3.
6 3652 4185 AT: 73.00 Bad overlap with gene 3.
7 4374 4604 AT: 77.00 Bad overlap with gene 3.
8 5622 6485 AT: 61.00
9 8621 7206 AT: 73.00
10 9601 9831 AT: 78.00
/predator/mpertea/MySoftware/GlimmerM/Malaria/

Figure 4.4.8 Sample output from the malaria-specific version of GlimmerM. The FASTA file used to generate this output is available on the Current Protocols Web site (*http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm*).

Use independent scores indicates whether the module that scores each fragment using independent base probabilities is used in the interpolated Markov model scores (see Background Information).

The parameters shown in Figure 4.4.8 have the default values, but they could have been changed if the optional parameters described in Table 4.4.1 were used. When GlimmerM is running, then it will use decision trees (see Background Information) in computing the splice-site scores only if these flags were created by the training procedure.

The next portion of the output in Figure 4.4.8 shows the scores of GlimmerM for each open reading frame in a given gene:

The first column represents an ID number for reference purposes. IDs are assigned to all predicted gene models.

Columns 2 to 7 are the scores for the coding region in each of the six reading frames. F1, F2, and F3 represent the three forward reading frames, while F4, F5, and F6 are the reverse complement frames. The score represents a normalized probability that the coding module (the IMM embedded in GlimmerM) generated the DNA sequence of the gene in that frame (for further details, see Salzberg et al., 1998a).

The `IndScore` column is the normalized probability that the DNA sequence was generated by a model of independent random probabilities for each base (Salzberg et al., 1998a). This is a simple model estimating the probability that the sequence is just random DNA with the same underlying GC composition.

The `Splice site scores` column shows the scores of the donor and acceptor splice sites of the potential gene models (Salzberg, 1997). Gene models with one intron have two scores shown (donor and acceptor, in order). Models with more introns have two scores shown per intron. Higher scores for the splice sites are better.

The last section of the output in Figure 4.4.8 begins with the phrase `Putative genes` and shows the gene models themselves. Each model begins with an ID number, followed by a list of exon positions beginning at the start codon and ending at the stop codon. Note that noncoding exons are not predicted at all, and that only the coding portions of the initial and terminal exons are shown. Thus, the initial "exon" begins with the translation start codon ATG, and the final exon ends at the position containing the stop codon for that gene. Previous studies showed that *P. falciparum* exons have an average AT-content of 70% to 75% (Salzberg et al., 1999), so GlimmerM also displays the A+T percentage for each exon. If a model overlaps another gene model that has a higher coding score or is simply longer, but either the score difference or the overlap length is small, then the message `Bad overlap with gene x` is printed, where `x` is the ID number of the better overlapping model.

Other Organisms

For all versions of GlimmerM except the malaria version, the system prints a list of the putative gene models (Fig. 4.4.9). The output is very similar to that produced by the GENSCAN system (Burge and Karlin, 1997); this format is intentionally designed to make it easier for software that parses the output of GENSCAN to use GlimmerM as well. For each gene model, the output contains a list of the exons that comprise that prediction. Four types of exons may appear in the predictions: initial (between a start codon and a donor site), internal (between two splice sites), terminal (between an acceptor site and a stop codon), and single (for unspliced genes). The exon length is also printed for reference. No probabilities are assigned to the exons, but this feature in a subsequent version of GlimmerM is planned to be implemented in order to permit the user to identify the best exons.

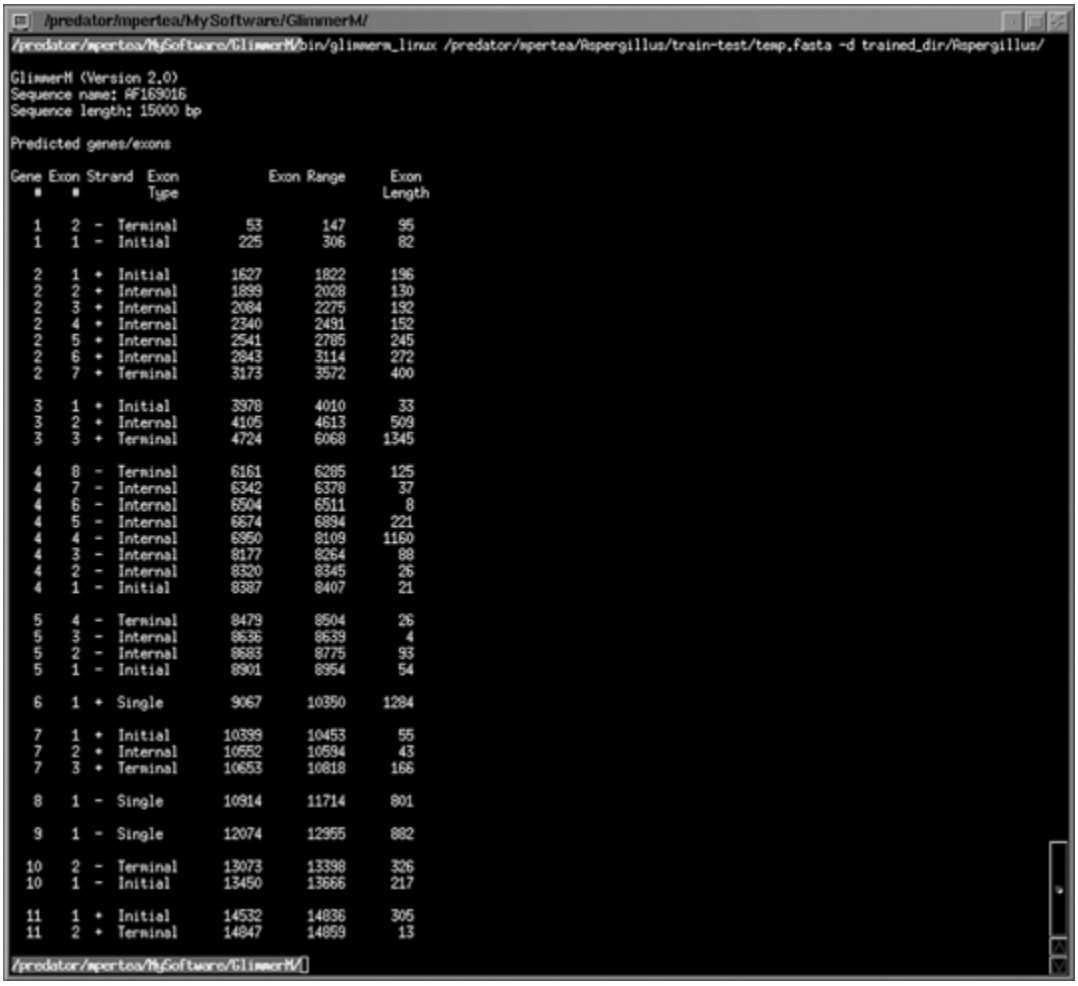

/predator/mpertea/MySoftware/GlimmerM/
/predator/mpertea/MySoftware/GlimmerM/bin/glimmerm_linux /predator/mpertea/Aspergillus/train-test/temp.fasta -d trained_dir/Aspergillus/
GlimmerM (Version 2.0)
Sequence name: AF169016
Sequence length: 15000 bp
Predicted genes/exons
Gene # Exon # Strand Exon Type Exon Range Exon Length
1 2 - Terminal 53 147 95
1 1 - Initial 225 306 82
2 1 + Initial 1627 1822 196
2 2 + Internal 1899 2028 130
2 3 + Internal 2084 2275 192
2 4 + Internal 2340 2491 152
2 5 + Internal 2541 2785 245
2 6 + Internal 2843 3114 272
2 7 + Terminal 3173 3572 400
3 1 + Initial 3978 4010 33
3 2 + Internal 4105 4613 509
3 3 + Terminal 4724 6068 1345
4 8 - Terminal 6161 6285 125
4 7 - Internal 6342 6378 37
4 6 - Internal 6504 6511 8
4 5 - Internal 6674 6894 221
4 4 - Internal 6950 8109 1160
4 3 - Internal 8177 8264 88
4 2 - Internal 8320 8345 26
4 1 - Initial 8387 8407 21
5 4 - Terminal 8479 8504 26
5 3 - Internal 8636 8639 4
5 2 - Internal 8683 8775 93
5 1 - Initial 8901 8954 54
6 1 + Single 9067 10350 1284
7 1 + Initial 10399 10453 55
7 2 + Internal 10552 10594 43
7 3 + Terminal 10653 10818 166
8 1 - Single 10914 11714 801
9 1 - Single 12074 12955 882
10 2 - Terminal 13073 13398 326
10 1 - Initial 13450 13666 217
11 1 + Initial 14532 14836 305
11 2 + Terminal 14847 14859 13
/predator/mpertea/MySoftware/GlimmerM/

Figure 4.4.9 Sample output of the current version of GlimmerM created by the Basic Protocol. The FASTA file used to generate this output is available on the Current Protocols Web site (*http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm*).

COMMENTARY

Background Information

Foundation and assumptions

The basis of GlimmerM is a dynamic-programming algorithm (*UNIT 3.1*) that considers all combinations of possible exons for inclusion in a gene model, and chooses the best of all these combinations. The possible exon-intron combinations are formed after an initial screening of the possible translational start sites and splice sites found in the genome. Both these entities are determined with specially designed modules based primarily on Markov chains. Markov models have been in use for decades as a method for modeling sequences. In particular, they have been remarkable for their success in modeling speech (Jelinek, 1997). Markov models are a natural way of modeling a sequence of events, and they translate very directly to DNA sequence data. Although other methods are in use, Markov models are among the most successful for finding genes in both prokaryotes and eukaryotes (Burge and Karlin, 1997; Salzberg et al., 1998a). To score a sequence using a Markov chain, a gene finder needs to compute a set of probabilities from training data. These probabilities take the form $P(b_i|b_{i-1},b_{i-2},b_{i-3}\ ...)$, where b_i indicates the base in position i of the DNA sequence. A 5^{th}-order Markov chain, for instance, would compute probabilities for each of the 4 bases following every possible 5-base combination; i.e., it would compute 4096 probabilities.

As with many other gene finders (Salzberg et al., 1998a), there are a number of assumptions used by GlimmerM to simplify the task

of gene prediction and narrow the possible choices when making predictions. The main assumptions are: (1) the coding region of every gene begins with a start codon ATG, (2) a gene has no in-frame stop codons and no frameshift mutations, (3) each exon is in a consistent reading frame with the previous exon, and (4) every intron begins and ends with the consensus dinucleotides GT-AG. These constraints significantly enhance the efficiency of the algorithm for searching through all possible gene models, by restricting the search space of the dynamic-programming algorithm. On the other hand, genuine frame shifts cannot be detected by the system.

Detecting splice sites

To detect splice sites in eukaryotic mRNA, GlimmerM combines several techniques that have already proven successful in characterizing the patterns around the donor and acceptor sites. The splice-site predictor algorithm uses a decision-tree method called maximal dependence decomposition (MDD), first introduced by Burge and Karlin (1997), which is enhanced by Markov models that capture additional dependencies among neighboring bases in a region around the splice site. This method considers only a small window around the splice junctions, which contains most of the information recognized by the spliceosome. The authors' algorithm also takes advantage of the fact that the coding and non-coding sequences switch at the splice junction, and this switch can sometimes be detected by considering sequence statistics in a larger window. In addition, by applying the local score optimality feature developed by Brendel and Kleffe (1998), the authors increased the overall performance of the splice-site detection system.

Using interpolated Markov models to select a gene model

Selecting the best gene model depends on a combination of the strength of the splice sites and the score of the exons produced by a special generalization of a Markov chain called an interpolated Markov model (IMM; Salzberg et al., 1999). IMMs are a generalization of fixed-order Markov chains. The main distinction is that rather than deciding in advance how many bases to consider for each prediction, these models will use varying numbers of bases for each prediction. In some contexts, they will use 5 bases, while in others they might use ≥6 bases, and yet in other cases, they may use ≤4 bases. This allows IMMs to be sensitive to how common a particular oligomer is in a given genome. In a given genome, many 5-mers might occur rarely and should not be used for prediction; here the IMM will fall back on a shorter Markov chain. On the other hand, certain 8-mers may occur very frequently, and for those, the IMM can use this longer context and make a better prediction. In addition, the IMM can combine the evidence from the 8th-order Markov chain and the 5th-order chain in such cases. Thus, it has all the information available to a 5th-order chain plus additional information. It is also worth noting that both IMMs and 5th-order Markov chains should outperform methods based on codon usage statistics.

GlimmerM uses the same IMM algorithm as the one described by Salzberg et al. (1998a) in the original Glimmer publication. IMMs form the basis of the Glimmer system, which finds genes in prokaryotes (bacteria, archaea, viruses) and in a few very simple eukaryotes (*T. brucei*). Glimmer correctly identifies ~99% of the genes in bacteria without any human intervention, and with a very limited number of false positives. Since its introduction, it has been used as the gene finder for *B. burgdorferi* (Fraser et al., 1997), *T. pallidum* (Fraser et al., 1998), *C. trachomatis* (Stephens et al., 1998), *T. maritima* (Nelson et al., 2001), *V. cholerae* (Heidelberg et al., 2000), and many other prokaryotes. Based on the success of Glimmer in bacterial sequence annotation, the authors hypothesized that IMMs would make a good foundation for eukaryotic gene finding. This is particularly true of small eukaryotes like *P. falciparum,* in which the gene density is intermediate between that of prokaryotes and higher eukaryotes.

To predict genes in malaria, GlimmerM runs separately over both the direct and complementary strands of the input. The algorithm then makes one more pass over the list of putative genes to reject overlapping genes. If genes overlap by less than a fixed amount (30 bp by default), then the overlap is ignored and both genes are reported in the output. Most overlapping genes are competing gene models that share a stop codon, with one or more alternative exons comprising the only differences between the models. Genes that overlap by >30 bp are re-scored using the IMM, and the one with the best score is retained. If the scores of two or more overlapping models differ from the maximum score by less than a small pre-set amount, then GlimmerM considers the scores equivalent and outputs all the models as possible genes. In these instances, it marks the longest

gene as the preferred one. Of course, the overlapping gene models predicted by GlimmerM may be an indication of alternative splicing—an often overlooked phenomena in the design of gene-recognition programs.

Incorporating other scoring systems

While the original GlimmerM system used only IMMs to score potential coding regions, the later versions of the system integrate several other scoring methods. Based on a method described in Salzberg et al. (1998b), the authors built a scoring function based on decision trees in order to estimate the probability that a DNA subsequence is coding or not. Five types of subsequences are evaluated: introns, initial exons, internal exons, final exons, and single exons. Each subsequence is run through ten different decision trees built with the OC1 system (Murthy et al., 1993, 1994). The probabilities obtained with the decision trees are averaged to produce a smoothed estimate of the probability that the given subsequence is of a certain type. In the end, as in the malaria version, a gene model is accepted only if the IMM score for the coding sequence in the correct reading frame exceeds a fixed threshold.

Applications of GlimmerM

GlimmerM was used as the primary gene finder for chromosome 2 of *P. falciparum* (Salzberg et al., 1999), and a later version was used to annotate two model plants *Arabidopsis thaliana* (The Arabidopsis Genome Initiative, 2000) and the still-ongoing project to sequence the rice genome, *Oryza sativa* (Yuan et al., 2001). It is currently being used at TIGR for annotation of the genomes of the parasite *Theileria parva* and the fungus *Aspergillus fumigatus*. Other projects are planned for the future.

Gene modeling performance

Next, some estimates of the accuracy of GlimmerM on selected organisms is provided. The main difficulty with training a gene finder for a newly sequenced genome is the lack of positive examples. Ideally, the training data set should contain genes that represent a random sample of genes in that genome, but practical considerations often make this requirement impossible to satisfy. The accuracy of the resulting gene finder will depend not only on the modeling technique used but also on the training set, and one can often dramatically improve gene finding results by re-training a system as more genes become known. Because of these aspects, the numbers presented here should be only considered as a rough estimate of the accuracy of GlimmerM.

The malaria-specific version of GlimmerM has an accuracy that has been measured on the two published *P. falciparum* chromosomes (Gardner et al., 1998; Bowman et al., 1999), using known genes from that organism in order to validate the accuracy. When computed at the nucleotide level, sensitivity and specificity are above 94% and 97%, respectively (Salzberg et al., 1999; Pertea et al., 2000); in other words, 94% of coding nucleotides are correctly labeled as coding. Using another measure of accuracy, GlimmerM predicted the precisely correct structure for 98 out of 113 genes (87%) on chromosome 2 of *P. falciparum*, and was able to predict partially correct models for 14 others. These numbers necessarily leave out any genes on that chromosome for which no independent evidence was available, and it is impossible to estimate a false-positive rate with the data available today.

The accuracy of GlimmerM has also been evaluated on the model plant *Arabidopsis thaliana,* and it is expected to be similar on *Oryza sativa* (Pertea and Salzberg, 2002). Table 4.4.3 presents the authors' results on the ARASET data collected by Pavy et al. (1999). ARASET contains 74 genomic sequence fragments with multiple genes in each sequence. In total, it contains 168 genes and 94 intergenic sequences. Note that although the accuracy of the predictions is high at the nucleotide and exon level, only 63 out of 168 are predicted perfectly. The main reason for the higher accuracy on *Plasmodium* as compared to *Arabidopsis* is the difference in gene structure. In malaria parasites (and many other single-celled eukaryotes), introns are short and few in number, and genes are relatively densely packed along the chromosomes. In plants and animals, genes have many more introns, the introns themselves are much longer, and the genes are sparsely distributed along the chromosomes, making them much harder to find.

Critical Parameters and Troubleshooting

Splice-site recognition

When designing the splice-site detection module of the gene finder, the authors only considered the problem of recognizing the highly conserved dinucleotides GT and AG at the 5′ and 3′ intron boundaries. While these dinucleotides are almost always present at the splice sites, others appear with lower frequency

Table 4.4.3 GlimmerM's Performance On ARASET, a Set of Genes from *Arabidopsis thaliana*

	Correct gene predictions	Correct start sites	Sn+Sp (nucleotide level)	Correct predicted exons/true exons
GlimmerM results	63	107	0.95+0.94	766/860

in at least some eukaryotic genomes. For example, the human genome contains a small number of AT-AC introns, which use a different set of splicing molecules. The GT-AG introns are spliced by a U2-type spliceosome, while the AT-AC introns are excised by a novel U12-type spliceosome (Dietrich et al., 1997). At least 11 distinct U12-type introns have been identified in *Arabidopsis thaliana* (Wu and Krainer, 1996), and many more are likely to be characterized in the future. *Arabidopsis* also contains a relatively high number of GC-AG introns, which use the normal splicing machinery but not the normal 5′ dinucleotide. The *Arabidopsis* data the authors used for training GlimmerM contained 2 AT-AC introns and 48 introns with GC-AG borders. Despite the inclusion of these in the training set, GlimmerM will miss these introns because it is designed to find only the standard GT-AG introns.

Setting thresholds for splice-site detection

Choosing appropriate thresholds for the routines that determine splice sites involves deciding upon a trade-off between false-negative and false-positive rates. This can be quite a challenging problem when only a small number of true sites are known (which is the usual situation). Invariably there are huge numbers of false-positive sites. Essentially, any GT dinucleotide that is not a known donor site can be used as a false site, and likewise for AG dinucleotides and donor sites. In choosing these thresholds, the user must decide how to maximize the specificity of the recognition task (i.e., the percentage of false sites that are correctly reject) without a big loss in sensitivity (i.e., the number of true sites that are found). One automated strategy is to set a threshold such that the system will always miss a fixed percentage of the true sites; another possibility is to set it so that none of the true sites are missed. The latter strategy usually results in a very high false-positive rate, which causes a serious degradation in the overall performance of the gene finder. Using a fixed percentage is more appealing, but the optimal percentage varies from one organism to another (Pertea et al., 2001). Sometimes the patterns around a true splice site may contain sequencing errors, yielding a true site that gets a very low score. As previously mentioned (Salzberg, 1997), maximizing the correlation coefficient (CC) is also a poor strategy for choosing the thresholds, in part because this statistic gives equal weight to positive and negative examples.

All the above considerations should be taken into account when setting the threshold. In the automated training protocol of GlimmerM (see Support Protocol), the threshold for calling a sequence a real splice site is chosen by examining the trade-off between the false positive and negative rates. The system creates a sorted list of thresholds, adjusting the scoring function so that it will miss 1,2,3, etc. true sites. For each of the associated false-negative values, it computes the false-positive rate. Typically, the false-positive rate drops very rapidly at first, as the low-scoring "true" sites are excluded from the calculation. With each successive removal of a true site, the false-positive rate falls further, but eventually the rate declines more slowly. The default threshold is chosen to be the score corresponding to the point at which the false-positive rate drops by <1%. To allow a greater flexibility in setting the signals' thresholds, the authors' training procedure allows the user to consult these false-positive and false-negative rates and reset the threshold to yield a different tradeoff (see Support Protocol).

Need for organism-specific training

GlimmerM trained for rice produces clearly superior gene models for rice sequences than GlimmerM trained for *Arabidopsis* (Pertea and Salzberg, 2002). A more thorough evaluation was done by running GlimmerM on 42 genes with EST or protein sequence homology that were extracted from rice BACs in the TIGR databases (*www.tigr.org/tdb/rice*). The *Arabidopsis* version of GlimmerM detected only 65% of the coding sequences of the genes, with a specificity of 86%, while the rice version detected 93% of the coding nucleotides with 90% specificity. More recent experience with *T. parva* further supports this observation. Using the *P. falciparum* version of the system produces vastly inferior predictions to a version trained on *T. parva* genes when analyzing *T. parva* data.

Although the currently trained versions of GlimmerM should work well on closely related organisms, the user should use the suggested training procedure to re-train the system for other eukaryotic organisms in order to improve the accuracy of the gene prediction. If very few genes are available, however, then the best course is to use a version of GlimmerM trained on the most closely related organism. For species on which GlimmerM is already trained, further improved performance by re-training the system as the amount of DNA sequences and continued growth of the number of validated genes from these species are expected.

Suggestions for Further Analysis

GlimmerMExon

Sometimes the DNA sequences that are being analyzed are nothing more than very short fragments, but scientists are still interested in finding any gene fragments on these sequences. If the input sequence contains only partial gene models, then GlimmerM might not predict anything because it is designed to identify only complete gene models, from start to stop. To address this issue, the authors have developed a new version of GlimmerM, called GlimmerMExon, which has the ability to predict partial genes. Genes that are missing either their 5′ or 3′ end (or both) can be recognized by this program. GlimmerMExon has thus far only been trained for *P. falciparum,* and its performance there is comparable to that of GlimmerM on DNA sequences containing at least one complete gene. Currently, this prototype lacks an automatic training procedure, but the authors plan to adapt GlimmerM's training procedure to work with GlimmerMExon in the near future.

Acknowledgements

This work was supported in part by the National Institutes of Health under grant R01-LM06845 and by the National Science Foundation under grants KDI-9980088 and IIS-9902923.

Literature Cited

Altschul, S., Gish, W., Miller, W., Myers, E., and Lipman, D. 1990. Basic local alignment search tool. *J. Mol. Biol*. 215:403-410.

Altschul, S., Madden, T., Schaffer, A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. 1997. Gapped blast and psi-blast: A new generation of protein database search programs. *Nucleic Acids Res.* 25:3389-3402.

The Arabidopsis Genome Initiative. 2000. Analysis of the genome sequence of the flowering plant *Arabidopsis thaliana*. *Nature* 408:796-815.

Bowman, S., Lawson, D., Basham, D., Brown, D., Chillingworth, T., Churcher, C.M., Craig, A., Davies, R.M., Devlin, K., Feltwell, T., Gentles, S., Gwilliam, R., Hamlin, N., Harris, D., Holroyd, S., Hornsby, T., Horrocks, P., Jagels, K., Jassal, B., Kyes, S., McLean, J., Moule, S., Mungall, K., Murphy, L., Barrell, B.G., et al. 1999. The complete nucleotide sequence of chromosome 3 of *Plasmodium falciparum*. *Nature* 400:532-538.

Brendel, V. and Kleffe, J. 1998. Prediction of locally optimal splice sites in plant pre-mRNA with applications to gene identification in *Arabidopsis thaliana* genomic DNA. *Nucleic Acids Res*. 26:4748-4757.

Burge, C. 1997. Ph.D. thesis. Identification of Genes in Human Genomic DNA. Standford University, Calif.

Burge, C.B. and Karlin, S. 1997. Prediction of complete gene structures in human genomic DNA. *J. Mol. Biol*. 268:78-94.

Dietrich, R.C., Incorvaia, R., and Padgett, R.A. 1997. Terminal intron dinucleotide sequences do not distinguish between U2- and U12-dependent introns. *Mol. Cell.* 1:151-160.

Florea, L., Hartzell, G., Zhang, Z., Rubin, G.M., and Miller, W. 1998. A computer program for aligning a cDNA sequence with a genomic DNA sequence. *Genome Res*. 8:967-974.

Fraser, C.M., Casjens, S., Huang, W.M., Sutton, G.G., Clayton, R., Lathigra, R., White, O., Ketchum, K.A., Dodson, R., Hickey, E.K., Gwinn, M., Dougherty, B., Tomb, J.F., Fleischmann, R.D., Richardson, D., Peterson, J., Kerlavage, A.R., Quackenbush, J., Salzberg, S., Hanson, M., van Vugt, R., Palmer, N., Adams, M.D., Gocayne, J., Venter, J.C., et al. 1997. Genomic sequence of a Lyme disease spirochaete, *Borrelia burgdorferi*. *Nature* 390:580-586.

Fraser, C.M., Norris, S.J., Weinstock, G.M., White, O., Sutton, G.G., Dodson, R., Gwinn, M., Hickey, E.K., Clayton, R., Ketchum, K.A., Sodergren, E., Hardham, J.M., McLeod, M.P., Salzberg, S., Peterson, J., Khalak, H., Richardson, D., Howell, J.K., Chidambaram, M., Utterback, T., McDonald, L., Artiach, P., Bowman, C., Cotton, M.D., Venter, J.C., et al. 1998. Complete genome sequence of *Treponema pallidum*, the syphilis spirochete. *Science* 281:375-388.

Gardner, M.J., Tettelin, H., Carucci, D.J., Cummings, L.M., Aravind, L., Koonin, E.V., Shallom, S., Mason, T., Yu, K., Fujii, C., Pederson, J., Shen, K., Jing, J., Aston, C., Lai, Z., Schwartz, D.C., Pertea, M., Salzberg, S., Zhou, L., Sutton, G.G., Clayton, R., White, O., Smith, H.O., Fraser, C.M., Hoffman, S.L., et al. 1998. Chromosome 2 sequence of the human malaria parasite *Plasmodium falciparum*. *Science* 282:1126-1132.

Heidelberg, J.F., Eisen, J.A., Nelson, W.C., Clayton, R.A., Gwinn, M.L., Dodson, R.J., Haft, D.H., Hickey, E.K., Peterson, J.D., Umayam, L., Gill, S.R., Nelson, K.E., Read, T.D., Tettelin, H., Richardson, D., Ermolaeva, M.D., Vamathevan, J., Bass, S., Qin, H., Dragoi, I., Sellers, P., McDonald, L., Utterback, T., Fleishmann, R.D., Nierman, W.C., and White, O. 2000. DNA sequence of both chromosomes of the cholera pathogen *Vibrio cholerae*. *Nature* 406:477-483.

Jelinek, F. 1997. Statistical Methods for Speech Recognition. MIT Press, Cambridge, MA.

Murthy, S.K., Kasif, S., Salzberg, S., and Beigel, R. 1993. OC1: Randomized induction of oblique decision trees. *Proc. 11th Natl. Conf. on Artificial Intelligence* 322-327.

Murthy, S.K., Kasif, S., and Salzberg, S. 1994. A system for induction of oblique decision trees. *J. of Artificial Intelligence Res.* 2:1-32.

Nelson, K.E., Eisen, J.A., and Fraser, C.M. 2001. Genome of *Thermotoga maritima* MSB8. *Methods Enzymol*. 330:169-180.

Pavy, N., Rombauts, S., Dehais, P., Mathe, C., Ramana, D.V., Leroy, P., and Rouze, P. 1999. Evaluation of gene prediction software using a genomic data set: Application to *Arabidopsis thaliana* sequences. *Bioinformatics* 15:887-899.

Pertea, M., Salzberg, S.L., and Gardner, M.J. 2000. Finding genes in *Plasmodium falciparum*. *Nature* 404:34-35.

Pertea, M. and Salzberg, S.L. 2002. Computational gene finding in plants. *Plant Molecular Biology* 48:9-48.

Pertea, M., Lin, X., and Salzberg, S.L. 2001. GeneSplicer: A new computational method for splice site prediction. *Nucleic Acids Res*. 29:1185-1190.

Salzberg, S.L. 1997. A method for identifying splice sites and translational start sites in eukaryotic mRNA. *Comput. Appl. Biosci*. 13:365-376.

Salzberg, S.L., Delcher, A.L., Kasif, S., and White, O. 1998a. Microbial gene identification using interpolated Markov models. *Nucleic Acids Res*. 26:544-548.

Salzberg, S., Delcher, A.L., Fasman, K.H., and Henderson, J. 1998b. A decision tree system for finding genes in DNA. *J. Comput. Biol*. 5:667-680.

Salzberg, S.L., Pertea, M., Delcher, A.L., Gardner, M.J., and Tettelin, H. 1999. Interpolated Markov models for eukaryotic gene finding. *Genomics* 59:24-31.

Stephens, R., Kalman, S., Lammel, C., Fan, J., Marathe, R., Aravind, L., Mitchell, W., Olinger, L., Tatusov, R., Zhao, Q., Koonin, E.V., and Davis, R.W. 1998. Genome sequence of an obligate intracellular pathogen of humans: *Chlamydia trachomatis*. *Science* 282:754-759.

Wu, Q. and Krainer, A.R. 1996. U1-mediated exon definition interactions between AT-AC and GT-AG introns. *Science* 274:1005-1008.

Yuan, Q., Quackenbush, J., Sultana, R., Pertea, M., Salzberg, S.L. and Buell, C.R. 2001. Rice bioinformatics: Analysis of rice sequence data and leveraging the data to other plant species. *Plant Physiol*. 125:1166-1174.

Key References

Salzberg et al., 1999. See above.

This paper introduces the GlimmerM method initially used in finding genes in Plasmodium falciparum. This paper also describes how GlimmerM was used in the annotation of chromosome 2 of P. falciparum.

Internet Resources

http://www.tigr.org/software/glimmerm/

GlimmerM Web site.

http://www.tigr.org/tdb/edb2/pfa1/htmls/

A preliminary annotation of chromosomes 10, 11, and 14 of P. falciparum. (This will change when the P. falciparum genome is completed.)

Contributed by Mihaela Pertea and
Steven L. Salzberg
The Institute for Genomic Research
Rockville, Maryland

Prokaryotic Gene Prediction Using GeneMark and GeneMark.hmm

In this unit, the GeneMark and GeneMark.hmm programs are presented as two different methods for the in silico prediction of genes in prokaryotes. GeneMark (see Basic Protocol 1), which uses Markov chain models and Bayes' rule (Durbin et al., 1998) to predict protein coding and noncoding regions, can be used for whole genome analysis as well as for the local analysis of a particular gene and its surrounding regions. GeneMark.hmm (see Alternate Protocol 1) makes use of hidden Markov models to find the transition points (boundaries) between protein coding states and noncoding states, and can be efficiently used for larger genome sequences. These methods can be used in conjunction with each other for a higher sensitivity of gene detection. They both include the option of using an RBS (ribosome binding site) model to aid in more accurate gene start prediction.

In either method, the number of genes accurately predicted depends on how well the sequence is described by the model. Models built from experimentally verified coding and noncoding regions are more reliable; however, the number of experimentally verified genes remains small, and thus other methods of building accurate models have been developed. The Heuristic Approach (see Basic Protocol 2) builds a fairly accurate inhomogeneous Markov model of protein coding regions based on the relationships between the positional nucleotide frequencies and the global nucleotide frequencies observed in the analysis of 17 complete bacterial genomes. This method can be used for sequences as small as 10 kbp. GeneMarkS (see Alternate Protocol 2) utilizes a nonsupervised training procedure and can be used for a newly sequenced prokaryotic genome with no prior knowledge of any protein or rRNA genes. However, it requires at least 1 Mbp of sequence. Models built from either of these procedures can then be used by GeneMark and GeneMark.hmm for sequence analysis.

The programs accessible through the Web site are periodically updated. Therefore, the users of the Web site always have access to the latest versions of the programs.

USING GeneMark FOR PROKARYOTIC GENE PREDICTION

BASIC PROTOCOL 1

The GeneMark program Web site (Fig. 4.5.1) uses precomputed statistical models for 32 species (as of August, 2002). A particular model is used to analyze a DNA sequence of the same organism. The algorithm of the GeneMark program and preliminary studies of the statistical models of DNA sequences have been described in several publications (Borodovsky et al., 1986a,b,c; Borodovsky and McIninch, 1993). The algorithm can determine a posteriori probabilities of protein coding in each of six possible frames for any given DNA sequence fragment. The output of the GeneMark program provides these a posteriori probability values as functions of sequence position. Then, an ORF (open reading frame) is identified as a gene based on the value of the a posteriori probability it accumulates. The text output of the program contains a list of predicted protein coding ORFs (Fig. 4.5.2). Optionally, the a posteriori probabilities for a given sequence can be viewed as a graph in six panels (Fig. 4.5.3). The default parameters of the program and the format of both the text and graphical outputs can be changed by the user.

A sample sequence (`example.fna`) is used below to illustrate how to use the GeneMark Web interface.

Contributed by Mark Borodovsky, Ryan Mills, John Besemer, and Alex Lomsadze
Current Protocols in Bioinformatics (2003) 4.5.1-4.5.16
Copyright © 2003 by John Wiley & Sons, Inc.

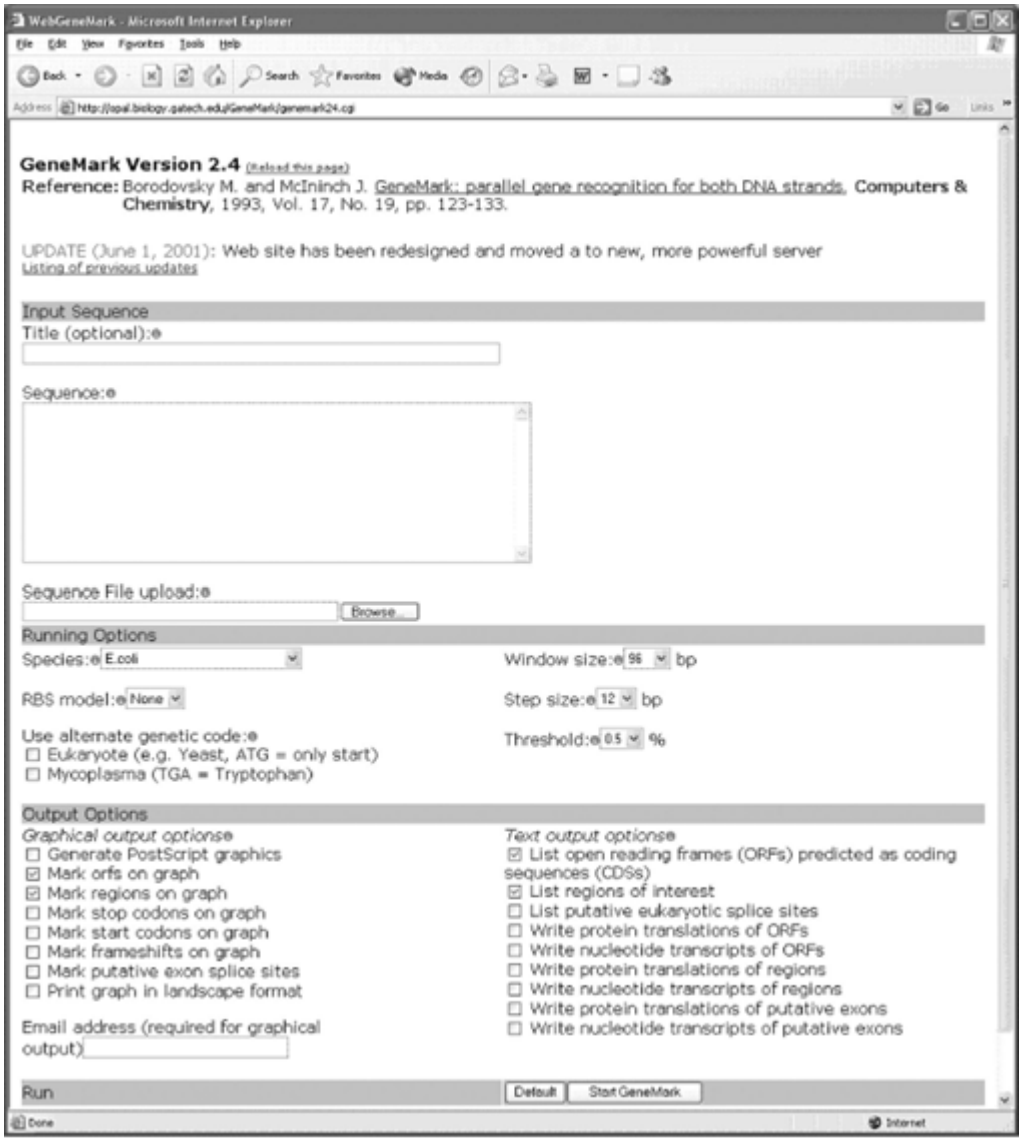

WebGeneMark - Microsoft Internet Explorer
GeneMark Version 2.4 (Reload this page)
Reference: Borodovsky M. and McIninch J. GeneMark: parallel gene recognition for both DNA strands. Computers & Chemistry, 1993, Vol. 17, No. 19, pp. 123-133.
UPDATE (June 1, 2001): Web site has been redesigned and moved a to new, more powerful server
Listing of previous updates
Input Sequence
Title (optional):
Sequence:
Sequence File upload:
Browse...
Running Options
Species: E.coli
Window size: 96 bp
RBS model: None
Step size: 12 bp
Use alternate genetic code:
Eukaryote (e.g. Yeast, ATG = only start)
Mycoplasma (TGA = Tryptophan)
Threshold: 0.5 %
Output Options
Graphical output options
Generate PostScript graphics
Mark orfs on graph
Mark regions on graph
Mark stop codons on graph
Mark start codons on graph
Mark frameshifts on graph
Mark putative exon splice sites
Print graph in landscape format
Email address (required for graphical output)
Text output options
List open reading frames (ORFs) predicted as coding sequences (CDSs)
List regions of interest
List putative eukaryotic splice sites
Write protein translations of ORFs
Write nucleotide transcripts of ORFs
Write protein translations of regions
Write nucleotide transcripts of regions
Write protein translations of putative exons
Write nucleotide transcripts of putative exons
Run
Default
Start GeneMark

Figure 4.5.1 The user interface for the GeneMark program. Required input includes a DNA sequence (either copied and pasted into the text box or uploaded as a file from the user's computer) in FASTA format, and the selection of the correct species model. Other options are available according to the interest of the individual user.

Necessary Resources

Hardware

A personal computer or workstation with Web access

Software

A Web browser

Files

A single sequence in FASTA format (*APPENDIX 1B*). The sample sequence (`example.fna`, which contains region 1 to 50,000 from *Escherichia coli* K12) used to illustrate this protocol can be downloaded from the *Current Protocols* Web site (*http://www3.interscience.wiley.com/c_p/cpbi_sample datafiles.htm*).

1. Via a Web browser, connect to *http://opal.biology.gatech.edu/GeneMark/genemark24.cgi*. In the Input Sequence section, paste an input sequence into the Sequence box area, or, alternatively, click on Browse next to the Sequence File Upload box to upload the input sequence file from a local drive.

   The Sequence File Upload option is more powerful, since the copy and paste method imposes a limit on the length of the sequence. If the sequence has a FASTA (APPENDIX 1B) title line (e.g., `> Sequence name`*), this name will be assigned to the sequence in the output, unless the user gives a name in the Sequence Title text area. For the purpose of analysis, all nonalphabet symbols are ignored and all ambiguous letters other than the symbols of the four nucleotides, assuming that they occur rarely, are replaced with* `C`*. This minimizes the chance of the possible creation of a false start or stop codon.*

2. Scroll down the page and set the Running Options. Select the name of the species of interest from the Species pull-down menu, which will result in the selection of the corresponding statistical model. The other pull-down menus, Window Size, Step Size, and Threshold, are set at default values chosen for being optimal in average results; however, the user has the option to change the default values. RBS models are available for some species. Models are available for some species with an alternative genetic code.

   Choosing the correct species name is essential, since wrong statistical models may totally corrupt the results of gene prediction. Sequences of species for which no model is available should be analyzed using either the heuristic method (see Basic Protocol 2) or GeneMarkS (see Alternate Protocol 2).

3. Scroll further down the page and select the Output Options. By default, the program generates text output in the form of a list of open reading frames predicted as coding sequences. The optional graphical output will be sent to the E-mail address provided by the user.

4. After completing the above entries, click the Start GeneMark button. The results will be depicted on the browser or will be sent to the E-mail address provided.

5. Interpret the text output. The GeneMark text output (Fig. 4.5.2) contains the following sections:

   a. Report header.

      Each report generated by GeneMark has a header confirming the parameters selected by the user in step 2 and indicating the name and order of the statistical model (matrix) used in the analysis.

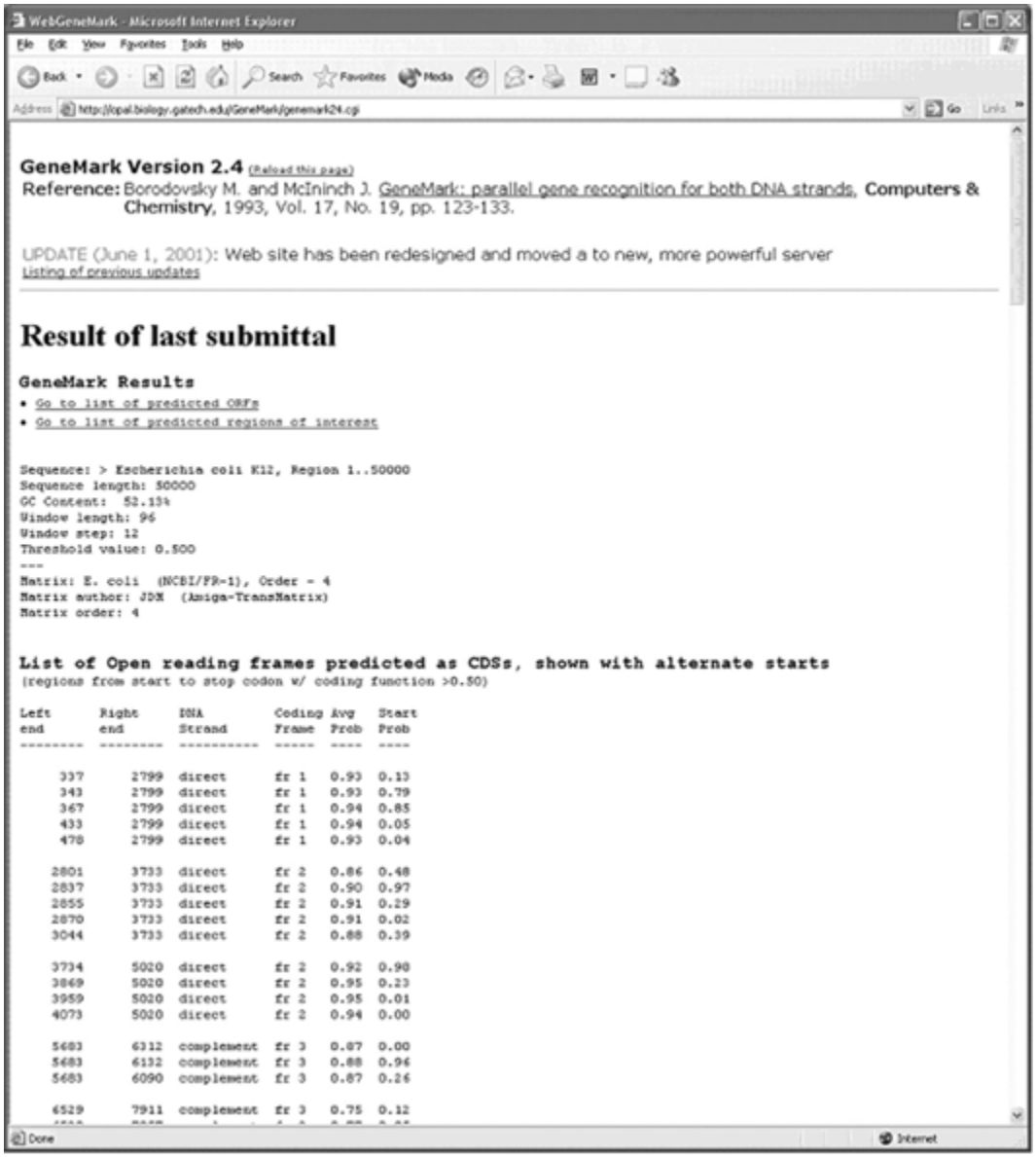
   b. List of Open Reading Frames.

      The sequence positions designated as `Left end` *and* `Right end` *define the boundaries of a predicted open reading frame relative to the sequence start (5′ end of the direct strand).* `DNA Strand` *indicates in which strand the coding region is located (direct or complement), and* `Coding Frame` *indicates the absolute reading frame. The* `Avg Prob` *column denotes the average coding potential over the indicated sequence range.*

      GeneMark does not indicate if an ORF extends beyond the limits of the sequence provided, so ORF end positions at 1,2,3 and at L-2,L-1,L (where L is the sequence length) may indicate that the ORF observed is just part of an ORF.

      The value shown in `Start Prob` *column is the likelihood that the start of the open reading frame is the actual start. For possible gene starts located closer than the window length to the sequence ends, this value is not calculated.*

      If an RBS model is specified, the program will predict putative RBS sites. The `RBS Prob` *value is a score indicating the likelihood of a particular oligomer (usually hexamer)*



WebGeneMark - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address http://opal.biology.gatech.edu/GeneMark/genemark24.cgi

GeneMark Version 2.4 (Reload this page)

Reference: Borodovsky M. and McIninch J. GeneMark: parallel gene recognition for both DNA strands. **Computers & Chemistry**, 1993, Vol. 17, No. 19, pp. 123-133.

UPDATE (June 1, 2001): Web site has been redesigned and moved a to new, more powerful server

Listing of previous updates

Result of last submittal

GeneMark Results

- Go to list of predicted ORFs
- Go to list of predicted regions of interest

```
Sequence: > Escherichia coli K12, Region 1..50000
Sequence length: 50000
GC Content:  52.13%
Window length: 96
Window step: 12
Threshold value: 0.500
---
Matrix: E. coli  (NCBI/FR-1), Order = 4
Matrix author: JDK  (Amiga-TransMatrix)
Matrix order: 4
```

List of Open reading frames predicted as CDSs, shown with alternate starts

(regions from start to stop codon w/ coding function >0.50)

Left end	Right end	DNA Strand	Coding Frame	Avg Prob	Start Prob
337	2799	direct	fr 1	0.93	0.13
343	2799	direct	fr 1	0.93	0.79
367	2799	direct	fr 1	0.94	0.85
433	2799	direct	fr 1	0.94	0.05
478	2799	direct	fr 1	0.93	0.04
2801	3733	direct	fr 2	0.86	0.48
2837	3733	direct	fr 2	0.90	0.97
2855	3733	direct	fr 2	0.91	0.29
2870	3733	direct	fr 2	0.91	0.02
3044	3733	direct	fr 2	0.88	0.39
3734	5020	direct	fr 2	0.92	0.90
3869	5020	direct	fr 2	0.95	0.23
3959	5020	direct	fr 2	0.95	0.01
4073	5020	direct	fr 2	0.94	0.00
5683	6312	complement	fr 3	0.87	0.00
5683	6132	complement	fr 3	0.88	0.96
5683	6090	complement	fr 3	0.87	0.26
6529	7911	complement	fr 3	0.75	0.12

Done

Internet

Figure 4.5.2 The text output from the GeneMark program for the example sequence. Open reading frames predicted as genes are listed for both strands, along with the average probability for an ORF and start probability for a translation start.

upstream to the putative start to be an RBS site. For each possible start, the program chooses the hexamer with the best score at a distance 4 to 21 nt from a putative start. The position of the rightmost (direct-strand) or leftmost (complement-strand) nucleotide of the hexamer with the best score and its sequence are shown. If the start site is adjacent to the edge of the sequence, it is not possible to evaluate the RBS site and no data are given. The minimum size of an ORF to be reported is half of the chosen window size parameter.

c. List of Regions of Interest.

GeneMark can identify so-called regions of interest, i.e., areas between in-frame stop codons with a high coding potential. The format of this list is similar to that for the open reading frames list (sample not shown).

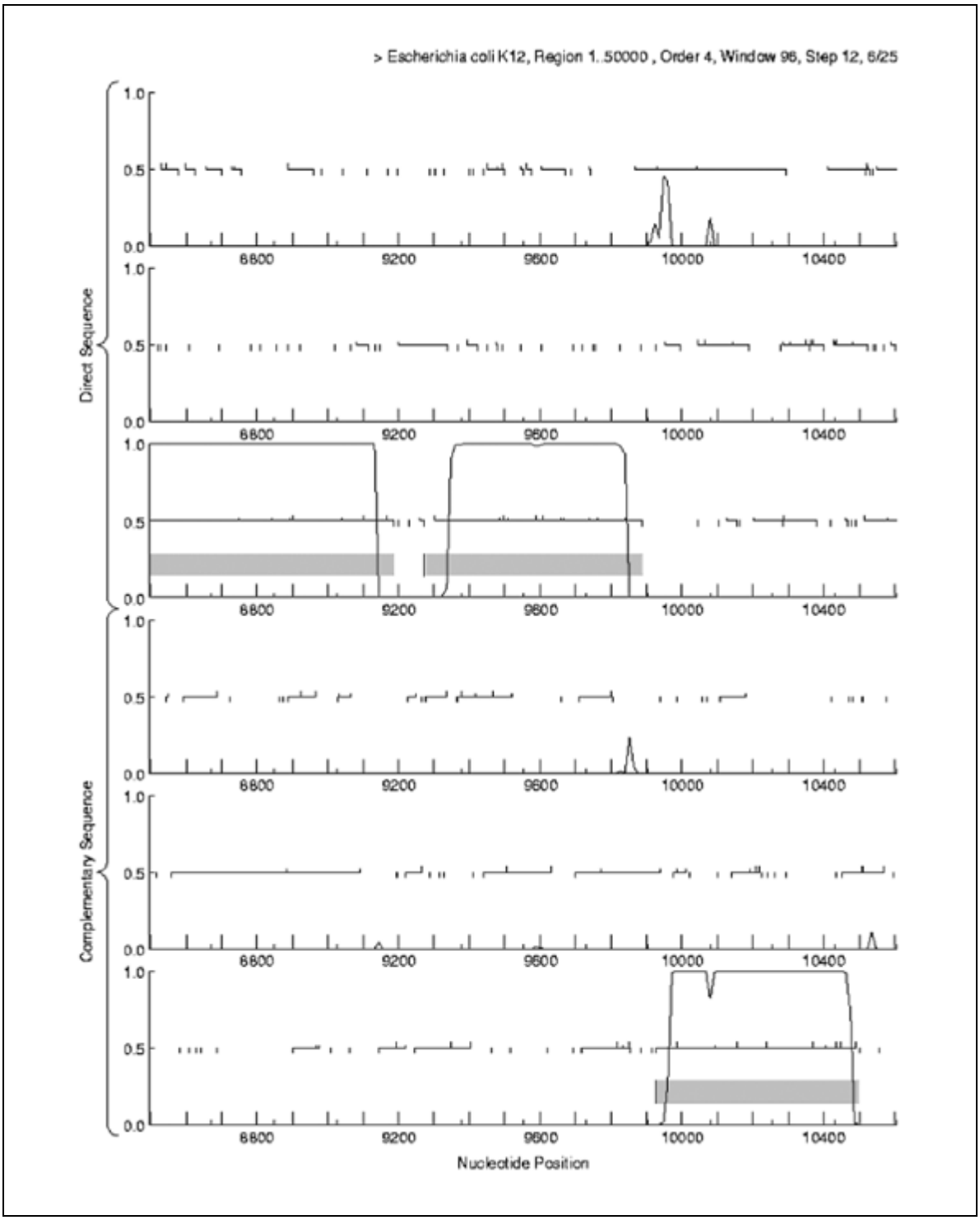


Figure 4.5.3 The graphical output from the GeneMark program for a region of the example sequence (`example.fna`). The six different panels represent the six possible reading frames, three each on the direct and reverse strands.

d. Detection of Possible Frameshifts.

GeneMark indicates possible frameshifts in protein-coding regions. A frameshift in the graphical output produces a switch of the coding potential graph from one panel to another panel related to the same DNA strand. This situation occurs when there is an insertion or deletion of one or several nucleotides (not a multiple of three) in a coding region. The table indicates the frame in which the codon region started, the frame in which the coding region continues, and the approximate location of the frameshift (the precision of which is determined by the Step Size parameter used) (sample not shown).

6. Interpret the graphical output.

The GeneMark graphical output (Fig. 4.5.3) depicts the coding potential in the six possible reading frames (three each on the direct and reverse strands). An unbroken horizontal line at the 0.5 level indicates an open reading frame (ORF). Large vertical ticks above the 0.5 level indicate "ATG" codons, while small vertical ticks represent "GTG." Vertical ticks under the 0.5 level represent one of the three stop codons, "TAA," "TGA," or "TAG." The thick gray horizontal line indicates a region of interest.

GeneMark generates the graphical output in Adobe PostScript format, which is sent by E-mail to the user. Some E-mail readers will allow the figure to be displayed automatically while others may require the installation of additional software programs. To create a valid PostScript file, one needs to select every line (inclusive) between `%!PS-Adobe-2.0` *and* `%%EOF` *and save it to a file called* `graph.ps` *or similar. The file should not contain any blank lines before the* `%!PS-Adobe-2.0` *line or after the* `%%EOF` *line. This file can then be viewed with any PostScript viewer program.*

ALTERNATE PROTOCOL 1

USING GeneMark.hmm FOR PROKARYOTIC GENE PREDICTION

The GeneMark.hmm program (Lukashin and Borodovsky, 1998; Besemer et al., 2001) uses a hidden Markov model framework with gene boundaries modeled as transitions between hidden states. In contrast to GeneMark (see Basic Protocol 1), the prediction of gene starts is further automated and human (expert) intervention to assess RBS sites and start codons is not required. GeneMark.hmm can be accessed through the Web (Fig. 4.5.4). It is convenient to use GeneMark.hmm for sequences with multiple genes, up to complete genomes where the speed of manual analysis becomes a bottleneck. GeneMark.hmm also uses precomputed models for 55 species (as of August, 2002). An additional convenience of GeneMark.hmm is the simultaneous running of Typical and Atypical models. Typical models are built from the major class of genes and Atypical models are built from the minor class, presumably laterally transferred genes. GeneMark (see Basic Protocol 1) can optionally be run simultaneously with GeneMark.hmm for these models as well. The output of the analysis can be presented in text (Fig. 4.5.5) and graphical (Fig. 4.5.6) formats.

The Typical models that are used with GeneMark.hmm have been constructed using the GeneMarkS program (see Alternate Protocol 2). The Atypical models have been built using the Heuristic model approach (see Basic Protocol 2). The Typical model is sensitive to the genes of the mainstream population sharing the same codon usage pattern. The Atypical model can find genes that do not match this common pattern, which often occurs with genes that have been horizontally transferred. Both models used in parallel find the vast majority of genes in a particular genome.

Necessary Resources

Hardware

A personal computer or workstation with Web access

Software

A Web browser

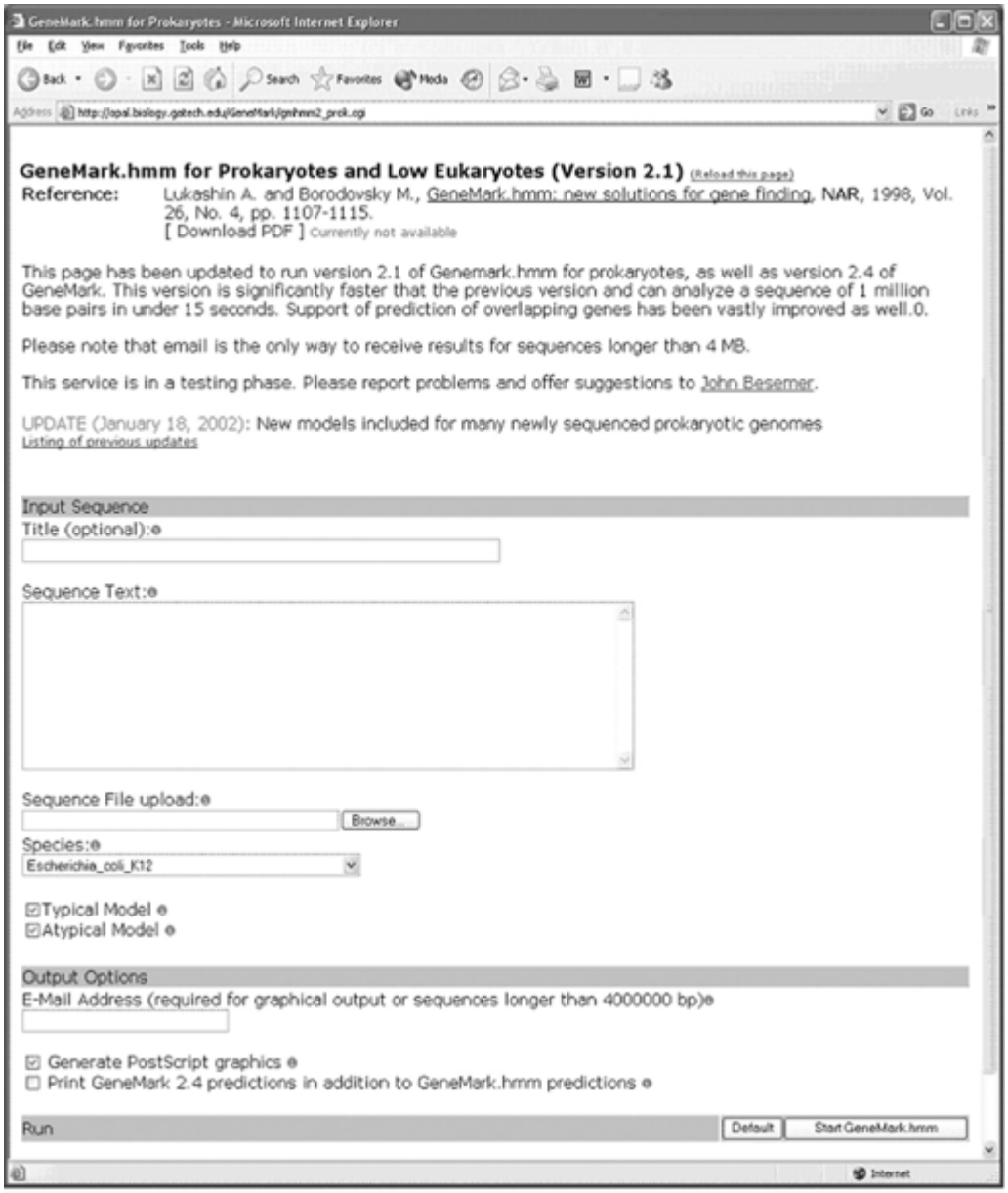

GeneMark.hmm for Prokaryotes - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://opal.biology.gatech.edu/GeneMark/gmhmm2_prok.cgi
Go Links
GeneMark.hmm for Prokaryotes and Low Eukaryotes (Version 2.1) (Reload this page)
Reference: Lukashin A. and Borodovsky M., GeneMark.hmm: new solutions for gene finding, NAR, 1998, Vol. 26, No. 4, pp. 1107-1115.
[Download PDF] Currently not available
This page has been updated to run version 2.1 of Genemark.hmm for prokaryotes, as well as version 2.4 of GeneMark. This version is significantly faster that the previous version and can analyze a sequence of 1 million base pairs in under 15 seconds. Support of prediction of overlapping genes has been vastly improved as well.0.
Please note that email is the only way to receive results for sequences longer than 4 MB.
This service is in a testing phase. Please report problems and offer suggestions to John Besemer.
UPDATE (January 18, 2002): New models included for many newly sequenced prokaryotic genomes
Listing of previous updates
Input Sequence
Title (optional):
Sequence Text:
Sequence File upload:
Browse...
Species:
Escherichia_coli_K12
Typical Model
Atypical Model
Output Options
E-Mail Address (required for graphical output or sequences longer than 4000000 bp)
Generate PostScript graphics
Print GeneMark 2.4 predictions in addition to GeneMark.hmm predictions
Run
Default
Start GeneMark.hmm
Internet

Figure 4.5.4 The user interface for the GeneMark.hmm program. Required input includes a DNA sequence (either copied and pasted into the text box or uploaded as a file from the user's computer) in FASTA format, the selection of the correct species model, and the selection of the Typical model, the Atypical model, or both.

Files

A single sequence in FASTA format (*APPENDIX 1B*). The sample sequence (`example.fna`, which contains region 1 to 50,000 from *Escherichia coli* K12) used to illustrate this protocol can be downloaded from the *Current Protocols* Web site (*http://www3.interscience.wiley.com/c_p/cpbi_sample datafiles.htm*).

1. Via a Web browser, connect to *http://opal.biology.gatech.edu/GeneMark/gmhmm2_prok.cgi*. In the Input Sequence section, paste an input sequence into the Sequence box area, or, alternatively, click on Browse next to the Sequence File Upload box to upload the input sequence file from a local drive.

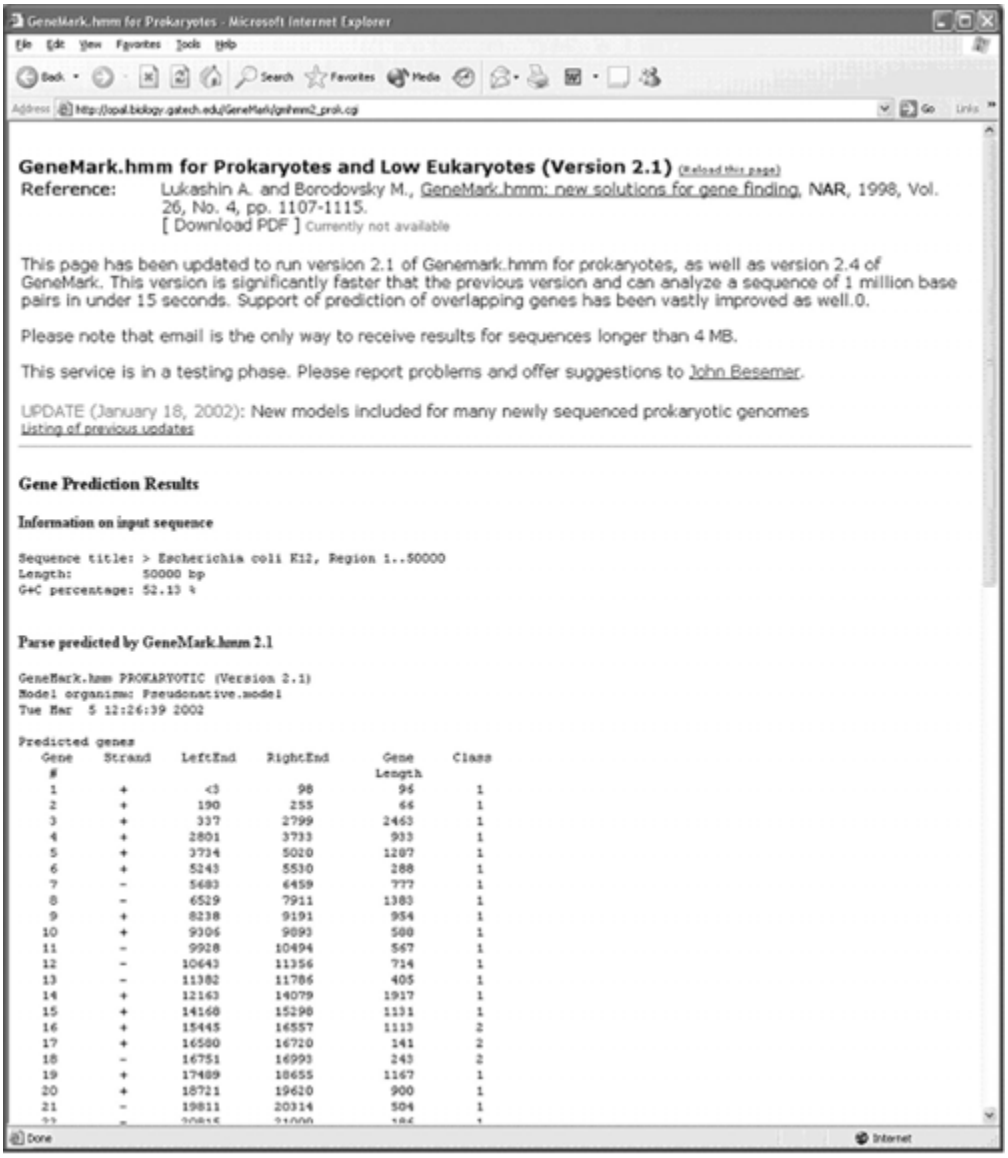

GeneMark.hmm for Prokaryotes - Microsoft Internet Explorer
GeneMark.hmm for Prokaryotes and Low Eukaryotes (Version 2.1)
Reference: Lukashin A. and Borodovsky M., GeneMark.hmm: new solutions for gene finding, NAR, 1998, Vol. 26, No. 4, pp. 1107-1115.
[Download PDF] Currently not available
This page has been updated to run version 2.1 of Genemark.hmm for prokaryotes, as well as version 2.4 of GeneMark. This version is significantly faster that the previous version and can analyze a sequence of 1 million base pairs in under 15 seconds. Support of prediction of overlapping genes has been vastly improved as well.0.
Please note that email is the only way to receive results for sequences longer than 4 MB.
This service is in a testing phase. Please report problems and offer suggestions to John Besemer.
UPDATE (January 18, 2002): New models included for many newly sequenced prokaryotic genomes
Listing of previous updates
Gene Prediction Results
Information on input sequence
Sequence title: > Escherichia coli K12, Region 1..50000
Length: 50000 bp
G+C percentage: 52.13 %
Parse predicted by GeneMark.hmm 2.1
GeneMark.hmm PROKARYOTIC (Version 2.1)
Model organism: Pseudonative.model
Tue Mar 5 12:26:39 2002
Predicted genes
Gene # Strand LeftEnd RightEnd Gene Length Class
1 + <3 98 96 1
2 + 190 255 66 1
3 + 337 2799 2463 1
4 + 2801 3733 933 1
5 + 3734 5020 1287 1
6 + 5243 5530 288 1
7 - 5683 6459 777 1
8 - 6529 7911 1383 1
9 + 8238 9191 954 1
10 + 9306 9893 588 1
11 - 9928 10494 567 1
12 - 10643 11356 714 1
13 - 11382 11786 405 1
14 + 12163 14079 1917 1
15 + 14160 15290 1131 1
16 + 15445 16557 1113 2
17 + 16580 16720 141 2
18 - 16751 16993 243 2
19 + 17489 18655 1167 1
20 + 18721 19620 900 1
21 - 19811 20314 504 1
Done
Internet

Figure 4.5.5 The text output from the GeneMark.hmm program for the example sequence using both the Atypical and Typical model. The predicted gene coordinates are listed along with the strand on which they were predicted, as well as the model class—1 for Typical and 2 for Atypical—from which it was predicted.

The Sequence File Upload option is more powerful, since the copy and paste method imposes a limit on the length of the sequence. If the sequence has a FASTA (APPENDIX 1B) title line (e.g., > Sequence name), this name will be assigned to the sequence in the output, unless the user gives a name in the Sequence Title text area. For the purpose of analysis, all nonalphabet symbols are ignored and all ambiguous letters other than the symbols of the four nucleotides, assuming that they occur rarely, are replaced with C*. This minimizes the chance of the possible creation of a false start or stop codon.*

2. Scroll down the page and select the name of the species of interest from the Species pull-down menu, which will result in the selection of the corresponding statistical model. By means of the two check boxes below the Species pull-down menu, the user may choose either the Typical model, the Atypical model, or both.

Choosing the correct species name is essential to obtaining meaningful results, since wrong statistical models may totally corrupt the results of gene prediction.

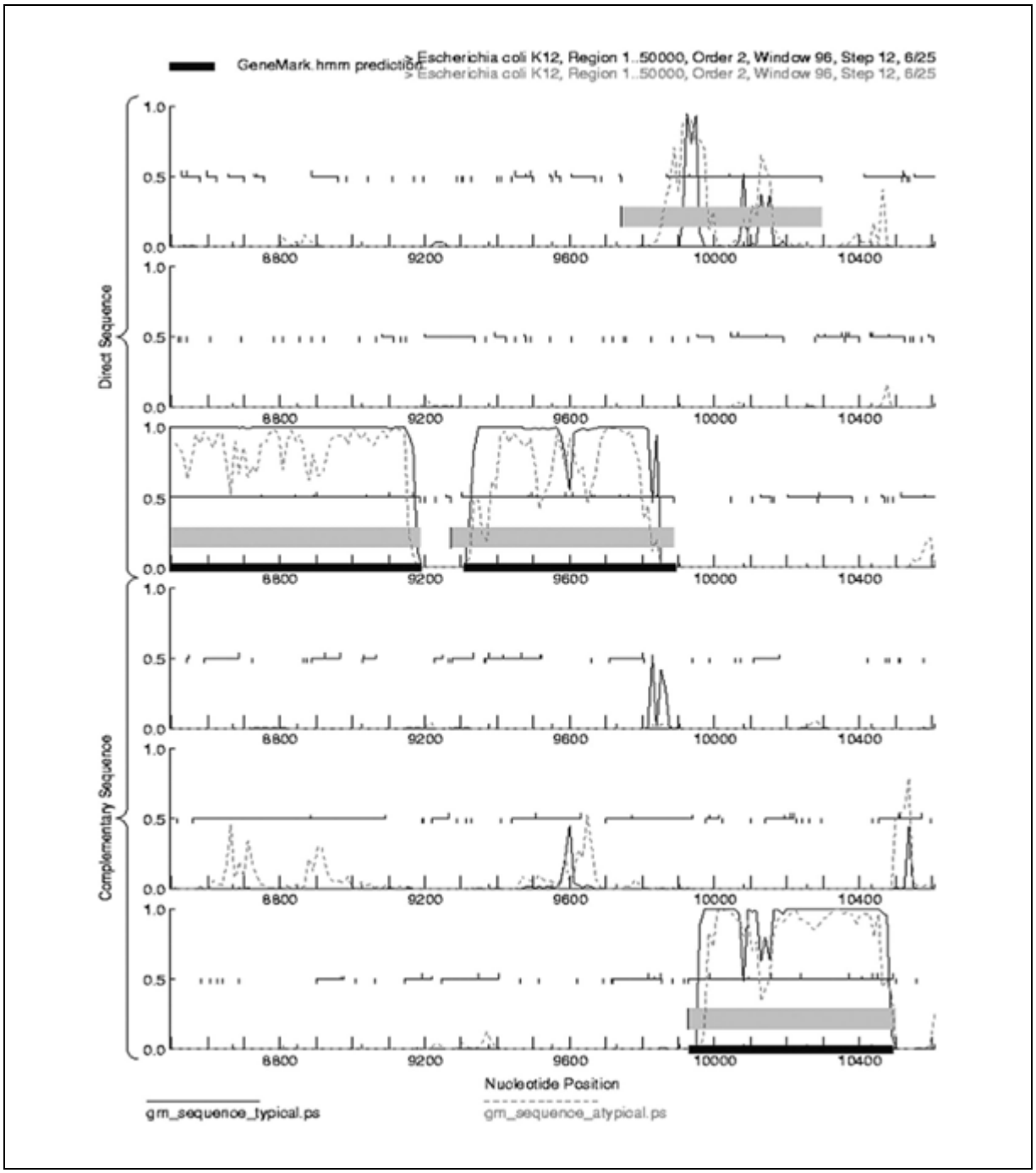


Figure 4.5.6 The graphical output from the GeneMark.hmm program for the sample sequence. The format is the same as the GeneMark graphical output, with the genes predicted by the Typical model in black (solid line) and the genes predicted by the Atypical model in red (dashed line). The wide black horizontal lines indicate regions predicted by GeneMark.hmm as protein coding.

3. Scroll further down the page and set the Output Options. The user may request the graphical output. Also, GeneMark predictions can be requested in addition to those of GeneMark.hmm. An E-mail address is required for sending text output, for sequences longer than 4,000,000 nt, or if graphical output is requested.
4. After completing the above entries, click the Start GeneMark.hmm button. The results will be depicted on the browser or will be sent to the E-mail address provided.

5. Interpret the text output.

   The GeneMark.hmm text output (Fig. 4.5.5) contains a listing of all regions predicted as protein coding. Genes predicted on the direct strand are indicated with a "+," while genes predicted on the complementary strand are indicated with a minus sign "−." If both Typical and Atypical models were selected, genes predicted by the Typical model will be denoted as Class 1 and genes predicted by the Atypical model as Class 2. If only one type of model was selected, only Class 1 will appear regardless of the model type. If an incomplete gene was predicted, a less than (<) or greater than (>) symbol will appear next to a coordinate for the left and right prediction edge, respectively.

6. Interpret the graphical output.

   The GeneMark.hmm graphical output is always combined with the GeneMark (see Basic Protocol 1) graphical output (Fig. 4.5.6). The predictions made by GeneMark.hmm are depicted by the thick black horizontal line on the axes of the six GeneMark panels. If both the Typical and Atypical models were selected, these horizontal lines will be shown in solid black and in dashed red fashions, respectively. If only one model was selected, the predictions will be shown by a solid black line.

BASIC PROTOCOL 2

USING THE HEURISTIC APPROACH FOR PROKARYOTIC MODEL BUILDING

The Heuristic algorithm (Besemer and Borodovsky, 1999), with the Web interface shown in Figure 4.5.7, builds an inhomogeneous Markov model of protein coding regions for an anonymous sequence based on the observed relationships between the positional nucleotide frequencies and the global nucleotide frequencies observed in the analysis of 17 complete bacterial genomes. This heuristic approach can be used for sequences as small as 10 kbp, making it especially useful in the analysis of small genomes, such as viruses and phages, for which there are not enough genes to build accurate models of higher order.

After creating a model from the sequence data, GeneMark.hmm and/or GeneMark will use the model just created to analyze the sequence.

Necessary Resources

Hardware

A personal computer or workstation with Web access

Software

A Web browser

Files

A single sequence in FASTA format (*APPENDIX 1B*). The sample sequence (`example.fna`, which contains region 1 to 50,000 from *Escherichia coli* K12) used to illustrate this protocol can be downloaded from the *Current Protocols* Web site (*http://www3.interscience.wiley.com/c_p/cpbi_sample datafiles.htm*).

1. Via a Web browser, connect to *http://opal.biology.gatech.edu/GeneMark/heuristic_hmm2.cgi*. In the Input Sequence section, paste an input sequence into the Sequence box area, or, alternatively, click on Browse next to the Sequence File Upload box to upload the input sequence file from a local drive.

   The Sequence File Upload option is more powerful, since the copy and paste method imposes a limit on the length of the sequence. If the sequence has a FASTA (APPENDIX 1B) title line (e.g., `> Sequence name`*), this name will be assigned to the sequence in the output, unless the user gives a name in the* `Sequence Title` *text area. For the purpose of analysis, all nonalphabet symbols are ignored and all ambiguous letters other than the*

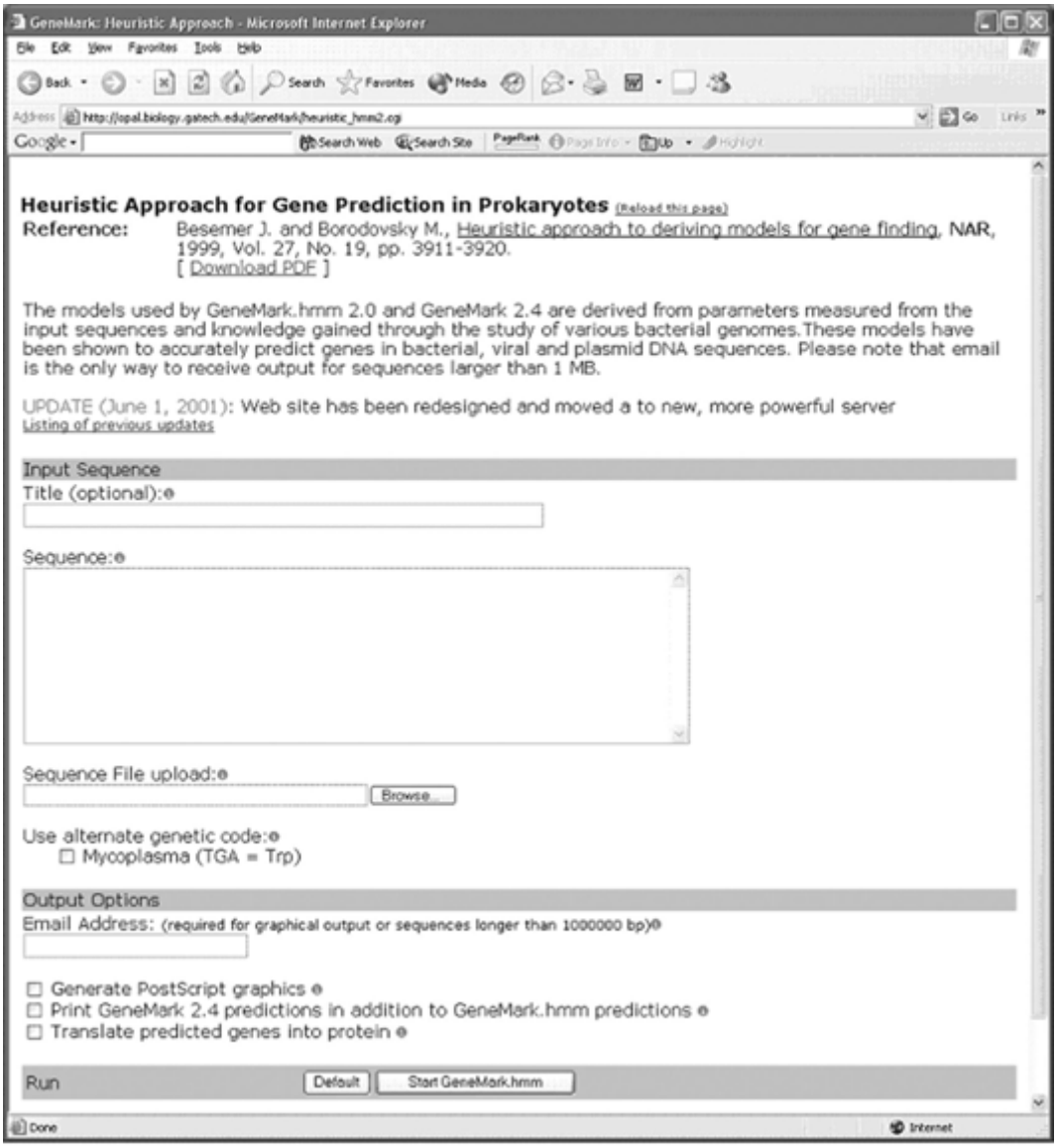

GeneMark: Heuristic Approach - Microsoft Internet Explorer
Heuristic Approach for Gene Prediction in Prokaryotes
Reference:
Besemer J. and Borodovsky M., Heuristic approach to deriving models for gene finding, NAR, 1999, Vol. 27, No. 19, pp. 3911-3920.
[Download PDF]
The models used by GeneMark.hmm 2.0 and GeneMark 2.4 are derived from parameters measured from the input sequences and knowledge gained through the study of various bacterial genomes. These models have been shown to accurately predict genes in bacterial, viral and plasmid DNA sequences. Please note that email is the only way to receive output for sequences larger than 1 MB.
UPDATE (June 1, 2001): Web site has been redesigned and moved a to new, more powerful server
Listing of previous updates
Input Sequence
Title (optional):
Sequence:
Sequence File upload:
Browse...
Use alternate genetic code:
Mycoplasma (TGA = Trp)
Output Options
Email Address: (required for graphical output or sequences longer than 1000000 bp)
Generate PostScript graphics
Print GeneMark 2.4 predictions in addition to GeneMark.hmm predictions
Translate predicted genes into protein
Run
Default
Start GeneMark.hmm
Done
Internet

Figure 4.5.7 The user interface for the Heuristic model building program. Required input includes a DNA sequence (either copied and pasted into the text box or uploaded as a file from the user's computer) in FASTA format. This program builds a model from the sequence, and then runs GeneMark.hmm.

symbols of the four nucleotides, assuming that they occur rarely, are replaced with C. *This minimizes the chance of the possible creation of a false start or stop codon.*

2. Scroll down the page to the Use Alternative Genetic Code check box, and if the species to be analyzed uses an alternative genetic code (*Mycoplasma* species), select this option.

3. Scroll further down the page and set the Output Options. The user may request the graphical output. Also, GeneMark predictions can be requested in addition to ones of GeneMark.hmm. An E-mail address is required for sending text output, for sequences longer than 1 Mbp, or if graphical output is requested.

4. After completing the above entries, click the Start GeneMark.hmm button. The results will be depicted on the browser or will be sent to the E-mail address provided.

5. Interpret the text output.

   The text output from the Heuristic approach is identical to that of GeneMark.hmm (see Alternate Protocol 1)

6. Interpret the graphical output.

   The graphical output from the Heuristic approach is identical to that of GeneMark.hmm (see Alternate Protocol 1) when using just one model.

ALTERNATE PROTOCOL 2

USING GeneMarkS FOR PROKARYOTIC MODEL BUILDING

The GeneMarkS program (Besemer et al., 2001), with the Web interface is shown in Figure 4.5.8, uses a nonsupervised training procedure to build a model for an anonymous DNA sequence. It does this through an iterative process beginning with analysis of the sequence using heuristic model building (see Basic Protocol 2).

Iteratively, GeneMarkS predicts coding regions and uses the predicted genes as a training set for the models used in the next iteration. The process ends when the predicted coding regions obtained with the updated models do not differ from the previous iteration. GeneMarkS also computes an RBS model using the Gibbs Motif Sampler (Lawrence et al., 1993) to align the upstream DNA sequences of the predicted starts.

GeneMarkS is efficient for the analysis of large genomes that possess enough coding and noncoding regions to derive accurate higher-order models. The use of the RBS model leads to a high accuracy of gene start predictions. For sequences smaller than 1 Mb, the accuracy may deteriorate.

GeneMarkS can also be used to analyze large phage genomes. Most phages use the translational machinery of their hosts. Translation signals such as the ribosomal binding site found in prokaryotes will thus be present in the phages that infect those organisms, and these signal sequences can consequently be detected using GeneMarkS in the same manner as for prokaryotes.

Necessary Resources

Hardware

A personal computer or workstation with Web access

Software

A Web browser

Files

A single sequence in FASTA format (*APPENDIX 1B*). The sample sequence (`example.fna`, which contains region 1 to 50,000 from *Escherichia coli* K12) used to illustrate this protocol can be downloaded from the *Current Protocols* Web site (*http://www3.interscience.wiley.com/c_p/cpbi_sample datafiles.htm*).

1. Via a Web browser, connect to *http://opal.biology.gatech.edu/GeneMark/genemarks.cgi*. In the Input Sequence section, paste an input sequence into the Sequence box area, or, alternatively, click on Browse next to the Sequence File Upload box to upload the input sequence file from a local drive.

   The Sequence File Upload option is more powerful, since the copy and paste method imposes a limit on the length of the sequence. If the sequence has a FASTA (APPENDIX 1B) title line (e.g., `> Sequence name`), this name will be assigned to the sequence in the output, unless the user gives a name in the Sequence Title text area. For the purpose of analysis, all nonalphabet symbols are ignored and all ambiguous letters other than the

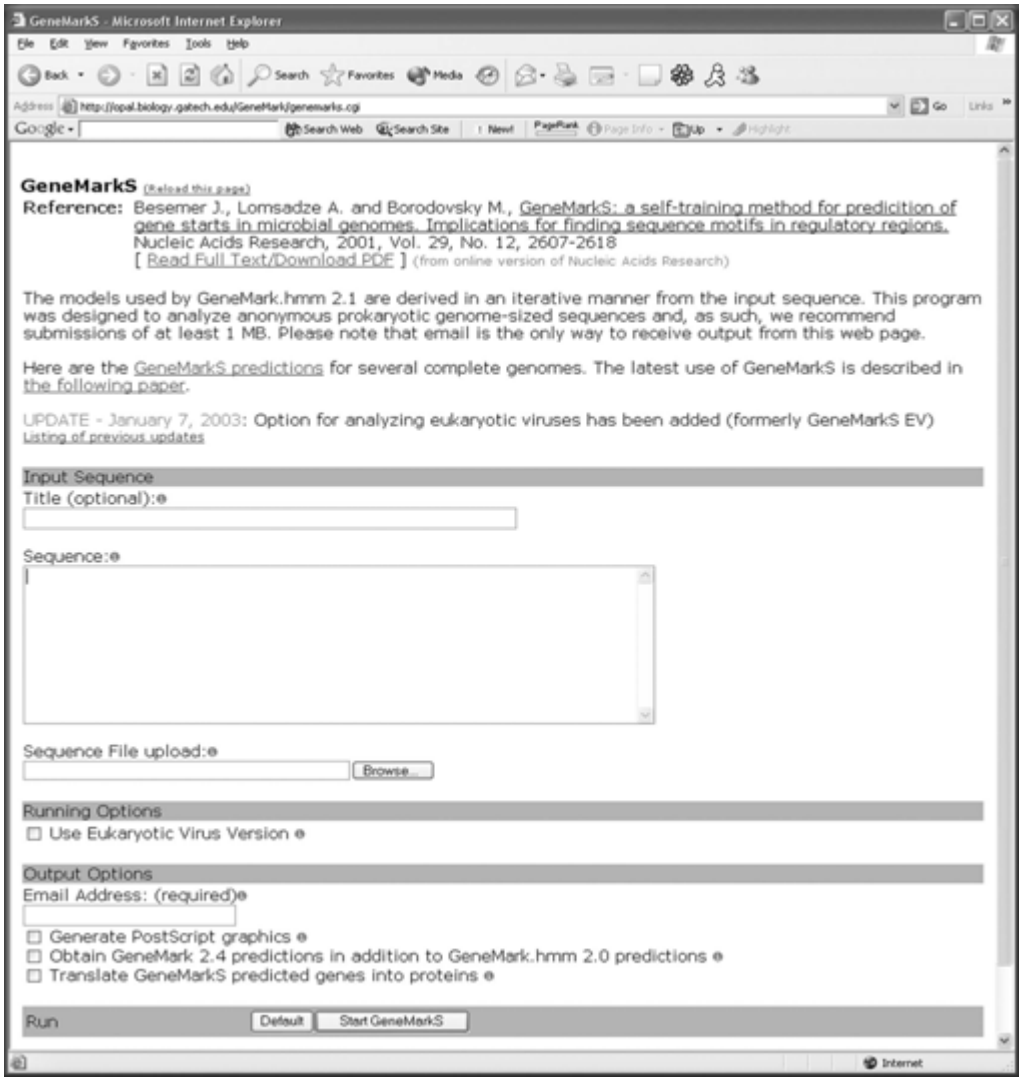

GeneMarkS - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
http://opal.biology.gatech.edu/GeneMark/genemarks.cgi
Go
Search Web Search Site
GeneMarkS (Reload this page)
Reference: Besemer J., Lomsadze A. and Borodovsky M., GeneMarkS: a self-training method for prediction of gene starts in microbial genomes. Implications for finding sequence motifs in regulatory regions. Nucleic Acids Research, 2001, Vol. 29, No. 12, 2607-2618
[Read Full Text/Download PDF] (from online version of Nucleic Acids Research)
The models used by GeneMark.hmm 2.1 are derived in an iterative manner from the input sequence. This program was designed to analyze anonymous prokaryotic genome-sized sequences and, as such, we recommend submissions of at least 1 MB. Please note that email is the only way to receive output from this web page.
Here are the GeneMarkS predictions for several complete genomes. The latest use of GeneMarkS is described in the following paper.
UPDATE - January 7, 2003: Option for analyzing eukaryotic viruses has been added (formerly GeneMarkS EV)
Listing of previous updates
Input Sequence
Title (optional):
Sequence:
Sequence File upload:
Browse...
Running Options
Use Eukaryotic Virus Version
Output Options
Email Address: (required)
Generate PostScript graphics
Obtain GeneMark 2.4 predictions in addition to GeneMark.hmm 2.0 predictions
Translate GeneMarkS predicted genes into proteins
Run
Default
Start GeneMarkS
Internet

Figure 4.5.8 The user interface for the GeneMarkS model building program. Required input includes a DNA sequence (either copied and pasted into the text box or uploaded as a file from the user's computer) in FASTA format. This program builds a model from the sequence, and then runs GeneMark.hmm. E-mail is the only way to receive output from this program.

symbols of the four nucleotides, assuming that they occur rarely, are replaced with C. *This minimizes the chance of the possible creation of a false start or stop codon.*

2. Scroll further down the page and set the Output Options. The user may request the graphical output. An E-mail address is required for sending text output, for sequences longer than 1 Mbp, or if graphical output is requested.
3. After completing the above entries, click the Start GeneMarkS button. The results will be depicted on the browser or will be sent to the E-mail address provided.
4. Interpret the text output.

   The text output from GeneMarkS is identical to that of GeneMark.hmm (see Alternate Protocol 1).

5. Interpret the graphical output.

   The graphical output from GeneMarkS is identical to that of GeneMark.hmm (see Alternate Protocol 1) when using just one model.

GUIDELINES FOR UNDERSTANDING RESULTS

The final steps of each protocol within this unit offers some guidance as to how to evaluate the results.

The output of the programs has a clear-cut meaning, i.e., the parsing of the DNA sequence into predicted coding and noncoding regions. Questions about the reliability of individual gene predictions have been addressed in previous publications (Borodovsky and McInInch, 1993; Lukashin and Borodovsky, 1998; Besemer and Borodovsky, 1999; Besemer et al., 2001).

COMMENTARY

Background Information

GeneMark

The GeneMark gene-prediction algorithm was developed in several steps. The first step was performed by a group at the Institute of Molecular Genetics in Moscow in 1986. In a series of three publications, it was demonstrated that inhomogeneous Markov chain models were useful tools for DNA sequence analysis and particularly for gene prediction (Borodovsky et al., 1986a,b,c). The GeneMark method itself was described in 1993 (Borodovsky and McInInch, 1993). Finding unnoticed genes in the *E. coli* DNA sequences by using GeneMark (Borodovsky et al., 1994a,b) served as solid evidence for the accurate predictive ability of the method. Another advantage of the method is its flexibility in a sense of rather easy training for a new species, or even for a separate class of genes within a given genome (Borodovsky et al., 1995). Application of the GeneMark program to help in interpreting the genomic sequences of *H. influenzae* and *M. genitalium* have taken advantage of these features of the algorithm (Fleischmann et al., 1995; Fraser et al., 1995; Tatusov et al., 1996).

The next step was necessary when completely new genomes (*M. jannaschii* and *H. pylori*) entered the scene with no experimentally studied segments that could be used for training (Bult et al., 1996; Tomb et al., 1997). At this point a new routine, called GeneMark-Genesis, was developed for parallel model learning and genomic sequence annotation. The models learned in this process could be diversified to accommodate the sets of so called Typical, Highly Typical, and Atypical genes that can be selected in a given bacterial genome (Hayes and Borodovsky, 1998a). For instance, in the case of the *E. coli* genome, the Highly Typical and Atypical genes correspond to Highly Expressed and Horizontally Transferred genes, respectively. The accuracy of gene prediction can be improved even further if an accurate model of the RBS signal is developed and taken into account (Hayes and Borodovsky, 1998b).

GeneMark.hmm

The GeneMark.hmm algorithm (Lukashin and Borodovsky, 1998) was designed to improve gene-prediction quality in terms of finding exact gene boundaries. The previously developed GeneMark program identified a gene mainly as the open reading frame where the gene is residing. However, the 5' boundary of the gene (the translation initiation codon associated with the protein amino terminus) might not be precisely predicted. The range of uncertainty for the initiation codon position is of the size of GeneMark sliding window, i.e., ~100 nt. In fact, GeneMark indicates several possible start codons and scores them. The underlying idea of GeneMark.hmm was to embed the GeneMark models for coding and noncoding regions into the naturally derived hidden Markov model (HMM) framework, with gene boundaries modeled as transitions between hidden states.

The HMM framework of GeneMark.hmm, i.e., the logic of transitions between hidden Markov states, followed the logic of the genetic structure of the bacterial genome. The Markov models of coding and noncoding regions were incorporated into the HMM framework to generate stretches of DNA sequence with coding or noncoding statistical patterns. This type of HMM architecture is known as "HMM with duration." The sequence of hidden states associated with a given DNA sequence carries information on positions where coding function is switching into noncoding and vice versa. The sequence of hidden states constitutes the HMM trajectory. The core GeneMark.hmm procedure, a dynamic programming–type algorithm, finds the most likely HMM trajectory given the DNA sequence. The newest version of Gene-

Mark.hmm (Besemer et al., 2001) has the capability of predicting genes with overlaps of arbitrary length. This version also integrates the two-component model of upstream conservative region (ribosomal binding site), the positional nucleotide frequency model, and spacer length distribution into the GeneMark.hmm algorithm.

Critical Parameters and Troubleshooting

The only program for which the users have the option to adjust the parameters is GeneMark. The main parameter of interest is the Threshold value, which determines the level of coding potential above which a region is considered a gene. For stringent gene finding (high specificity), a higher threshold is recommended. For a larger number of predictions (high sensitivity), a lower threshold should be used. By default, the threshold is set at 0.5.

The Window Size default value is 96.

The Step Size by which the sliding window moves is equal to 12 nt, as a default.

Problems may arise if the format is not a correct one (FASTA). Sequences that are too large (i.e., larger than 5 Mbp) should be split into smaller sequences for analysis. Sequences that are too small (i.e. smaller than 400 bp) cannot be accurately analyzed and should not be submitted.

Suggestions for Further Analysis

The protein translations of the predicted genes can be easily used in BLASTP (*UNIT 3.3*) to obtain additional information about the putative protein. Experimental biologists can use the sequences around the predicted genes to create primers for PCR analysis of genes of interest, as well as for designing DNA expression arrays.

Literature Cited

Besemer, J. and Borodovsky, M. 1999. Heuristic approach to deriving models for gene finding. *Nucleic Acids Res.* 27:3911-3920.

Besemer, J., Lomsadze, A., and Borodovsky, M. 2001. GeneMarkS: A self-training method for prediction of gene starts in microbial genomes: Implications for finding sequence motifs in regulatory regions. *Nucleic Acids Res.* 29:2607-2618.

Borodovsky, M. and McIninch, J. 1993. GENMARK: Parallel gene recognition for both DNA strands. *Comput. Chem.* 17:123-133.

Borodovsky, M., Sprizhitsky Yu., Golovanov, E., and Alexandrov, A. 1986a. Statistical patterns in primary structures of functional regions in the *E. coli* genome: I. Oligonucleotide frequencies analysis. *Mol. Biol.* 20:826-833.

Borodovsky, M., Sprizhitsky, Yu, Golovanov, E., and Alexandrov, A. 1986b. Statistical patterns in primary structures of functional regions in the *E. coli* genome: II. Non-homogeneous Markov models. *Mol. Biol.* 20:833-840.

Borodovsky, M., Sprizhitsky, Yu., Golovanov, E., and Alexandrov, A. 1986c. Statistical patterns in primary structures of functional regions in the *E. coli* genome: III. Computer recognition of coding regions. *Mol. Biol.* 20:1145-1150.

Borodovsky, M., Rudd, K., and Koonin, Eu. 1994a. Intrinsic and extrinsic approaches for detecting genes in a bacterial genome. *Nucleic Acids Res.* 22:4756-4767.

Borodovsky, M., Koonin, Eu., and Rudd, K. 1994b. New genes in old sequences: A strategy for finding genes in a bacterial genome. *Trends Biochem. Sci.* 19:309-313.

Borodovsky, M., McIninch, J., Koonin, E., Rudd, K., Medigue, C. and Danchin, A. 1995. Detection of new genes in the bacterial genome using Markov models for three gene classes. *Nucleic Acids Res.* 23:3554-3562.

Bult, C.J., White, O., Olsen, G.J., Zhou, L., Fleischmann, R.D., Sutton, G.G., Blake, J.A., FitzGerald, L.M., Clayton, R.A., Gocayne, J.D., Kerlavage, A.R.., Dougherty, B.A., Tomb, J.-F., Adams, M.D., Reich, C.I., Overbeek, R., Kirkness, E.F., Weinstock, K.G., Merrick, J.M., Glodek, A., Scott, J.L., Geoghagen, N.S.M., Weidman, J.F., Fuhrmann, J.L., Nguyen, D., Utterback, T.R., Kelley, J.M., Peterson, J.D., Sadow, P.W., Hanna, M.C., Cotton, M.D., Roberts, K.M., Hurst, M.A., Kaine, B.P., Borodovsky, M., Klenk, H.-P., Fraser, C.M., Smith, H.O., Woese, C.R., Venter, J.C. 1996. Complete genome sequence of the methanogenic archaeon *Methanococcus jannaschii*. *Science* 273:1058-1073

Durbin, R., Eddy, S., Krough, A., and Mitchison, G. 1998. Biological sequence analysis: Probabilistic models of proteins and nucleic acids. Cambridge University Press, Cambridge.

Fleischmann, R.D., Adams, M.D., White, O., Clayton, R.A., Kirkness, E.F., Kerlavage, A.R., Bult, C.J., Tomb, J.-F., Dougherty, B.A., Merrick, J.M., McKenney, K., Sutton, G., Fitzhugh, W., Fields, C.A., Gocayne, J.D., Scott, J.D., Shirley, R., Liu, L.-I., Glodek, A., Kelley, J.M., Weidman, J.F., Phillips, C.A., Spriggs, T., Hedblom, E., Cotton, M.D., Utterback, T.R., Hanna, M.C., Nguyen, D.T., Saudek, D.M., Brandon, R.C., Fine, L.D., Fritchman, J.L., Fuhrmann, J.L., Geoghagen, N.S.M., Gnehm, C.L., McDonald, L.A., Small, K.V., Fraser, C.M., Smith, H.O., and Venter, J.C. 1995. Whole-genome random sequencing and assembly of *Haemophilus influenzae* Rd. *Science* 269:496-512.

Fraser, C.M., Gocayne, J.D., White, O., Adams, M.D., Clayton, R.A., Fleischmann, R.D., Bult, C.J., Kerlavage, A.R., Sutton, G., Kelley, J.M., Fritchman, J.L., Weidman, J.F., Small, K.V., Sandusky, M., Fuhrmann, J.L., Nguyen, D.T., Utter-

back, T.R., Saudek, D.M., Phillips, C.A., Merrick, J.M., Tomb, J.-F., Dougherty, B.A., Bott, K.F., Hu, P.-C., Lucier, T.S., Peterson, S.N., Smith, H.O., Hutchison, III, C.A., and Venter, J.C. 1995. The minimal gene complement of *Mycoplasma genitalium*. *Science* 270:397-403.

Hayes, W. and Borodovsky, M. 1998a. How to interpret anonymous genome? Machine learning approach to gene identification. *Genome Res.* 8:1154-1171.

Hayes, W.S. and Borodovsky, M. 1998b. Deriving Ribosome Binding Site (RBS) statistical models from unannotated DNA sequences and the use of the RBS model for N-terminal prediction. *In* Proceedings of Pacific Symposium on Biocomputing 1998 pp. 279-290. World Scientific Press, Singapore.

Lawrence, C.E., Altschul, S.F., Boguski, M.S., Liu, J.S., Neuwald, A.F. and Wootton, J.C 1993. Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment. *Science* 262:208-214.

Lukashin, A.V. and Borodovsky, M. 1998. GeneMark.hmm: New solutions for gene finding. *Nucleic Acids Res.* 26:1107-1115.

Tatusov, R.L., Mushegian, A.R., Bork, P., Brown, N.P., Hayes W., Borodovsky, M., Rudd, K.E. and Koonin, E.V. 1996. Metabolism and evolution of *H. influenzae* deduced from whole genome comparison to *E. coli*. *Curr. Biol.* 6:279-291.

Tomb, J., White, O., Kerlavage, A.R., Clayton, R.A., Sutton, G.G., Fleischmann, R.D., Ketchum, K.A., Klenk, H.P., Gill, S., Dougherty, B.A., Nelson, K., Quackenbush, J., Zhou, L., Kirkness, E.F., Peterson, S., Loftus, B., Richarson, D., Dodson, R., Khalak, H.G., Glodek, A., McKenney, K., Fitzegerald, L.M., Lee, N., Adams, M.D., Hickey, E.K., Berg, D.E., Gocayne, J.D., Utterback, T.R., Peterson, J.D., Kelley, J.M., Cotton, M.D., Weidman, J.M., Fujii, C., Bowman, C., Watthey, L., Wallin, E., Hayes, W.S., Borodovsky, M., Karp, P.D., Smith, H.O., Fraser, C.M., and Venter, J.C. 1997. The complete genome sequence of the gastric pathogen *Helicobacter pylori*. *Nature* 388:539-547

Internet Resources

http://opal.biology.gatech.edu/GeneMark

GenMark Web site

Contributed by Mark Borodovsky
School of Biology and School of Biomedical Engineering
Georgia Institute of Technology
Atlanta, Georgia

Ryan Mills, John Besemer, and Alex Lomsadze
School of Biology
Georgia Institute of Technology
Atlanta, Georgia

Eukaryotic Gene Prediction Using GeneMark.hmm

In this unit, eukaryotic GeneMark.hmm (Lukashin and Borodovsky, 1998; M. Borodovsky and A.V. Lukashin, unpub. observ.) is presented as a method for detecting genes in eukaryotic DNA sequences. Detection of genes in sequences from prokaryotic organisms by GeneMark.hmm is described in *UNIT 4.5*. The eukaryotic GeneMark.hmm uses Markov models of protein coding and noncoding sequences, as well as positional nucleotide frequency matrices for prediction of the translational start, translational termination, and splice sites. All these models, along with length distributions of exons, introns, and intergenic regions, are integrated into one hidden Markov model. The algorithm implemented in GeneMark.hmm finds the maximum likelihood path of this model through hidden coding and noncoding states given the analyzed sequence. The GeneMark program (Borodovsky and McIninch, 1993; also see *UNIT 4.5*) may be run in conjunction with GeneMark.hmm to provide additional insight into how the DNA sequence is structured in terms of coding potential.

These GeneMark.hmm and GeneMark programs are accessible via the Internet (see Basic Protocol) at *http://opal.biology.gatech.edu/GeneMark/*. Alternatively, local versions of the software are available, which can be run under the Unix operating system (see Alternate Protocol 1). The programs on the Web site are revised as soon as updates are available; therefore, the users of the Web site normally have access to the latest version of the software.

A modified version of GeneMarkS (Besemer et al., 2001) can be used to detect genes in eukaryotic viruses (see Alternative Protocol 2). This program utilizes a nonsupervised training procedure and can be used with no prior knowledge of any protein. However, the current version of the program does not find introns, which may occur, though rarely, in viral genes.

USING WEB-INTERFACE GeneMark.hmm FOR EUKARYOTIC GENE PREDICTION

BASIC PROTOCOL

The Web site shown in Figure 4.6.1 is an interface for the Eukaryotic GeneMark.hmm program, which is run on an IBM RS/6000 server at the School of Biology of the Georgia Institute of Technology. The gene prediction results are reported as a list of exon coordinates (Fig. 4.6.2). Optionally, graphical output and a list of predicted protein sequences can be produced.

This protocol describes GeneMark.hmm version 2.2 and GeneMark version 2.4.

Necessary Resources

Hardware

A personal computer or workstation with Web access

Software

A Web browser, e.g., Netscape Communicator or Microsoft Internet Explorer

Files

A single sequence in FASTA format (see *APPENDIX 1B* and Pearson, 1990). The sample sequence—DNA sequence of complete human serum albumin (*ALB*) gene (GenBank accession no. M12523)—used to illustrate this protocol can be

Contributed by Mark Borodovsky, Alex Lomsadze, Nikolai Ivanov, and Ryan Mills
Current Protocols in Bioinformatics (2003) 4.6.1-4.6.12
Copyright © 2003 by John Wiley & Sons, Inc.

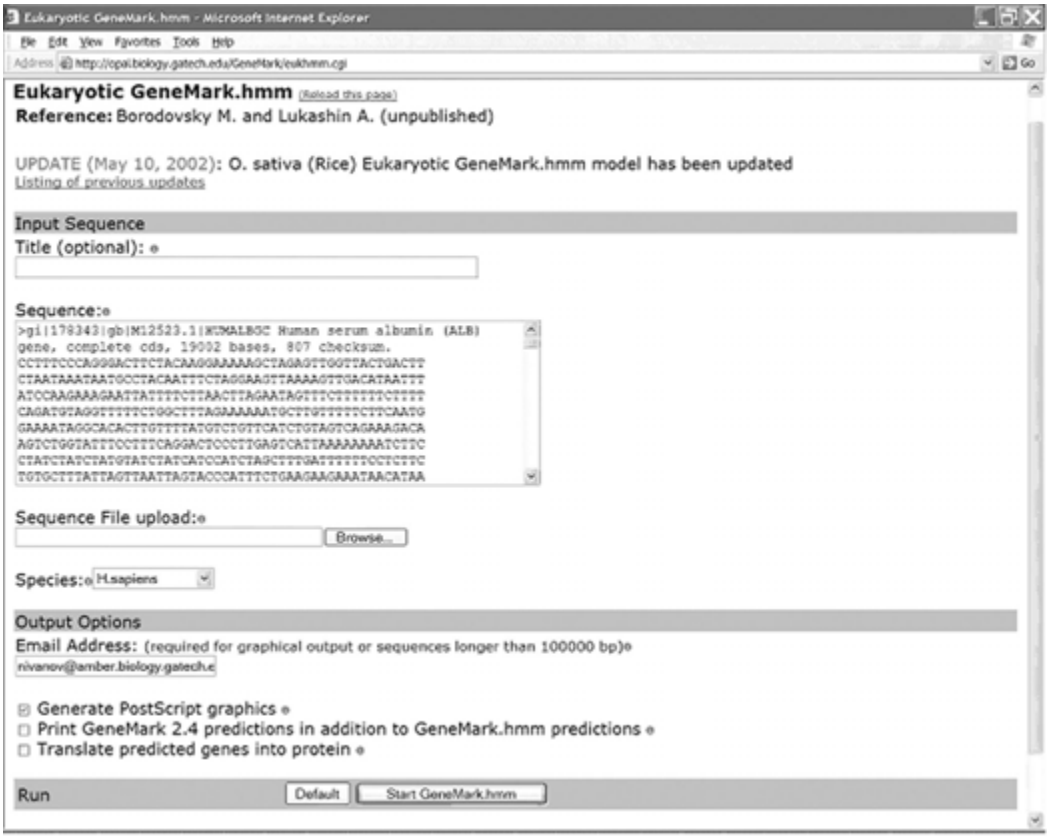
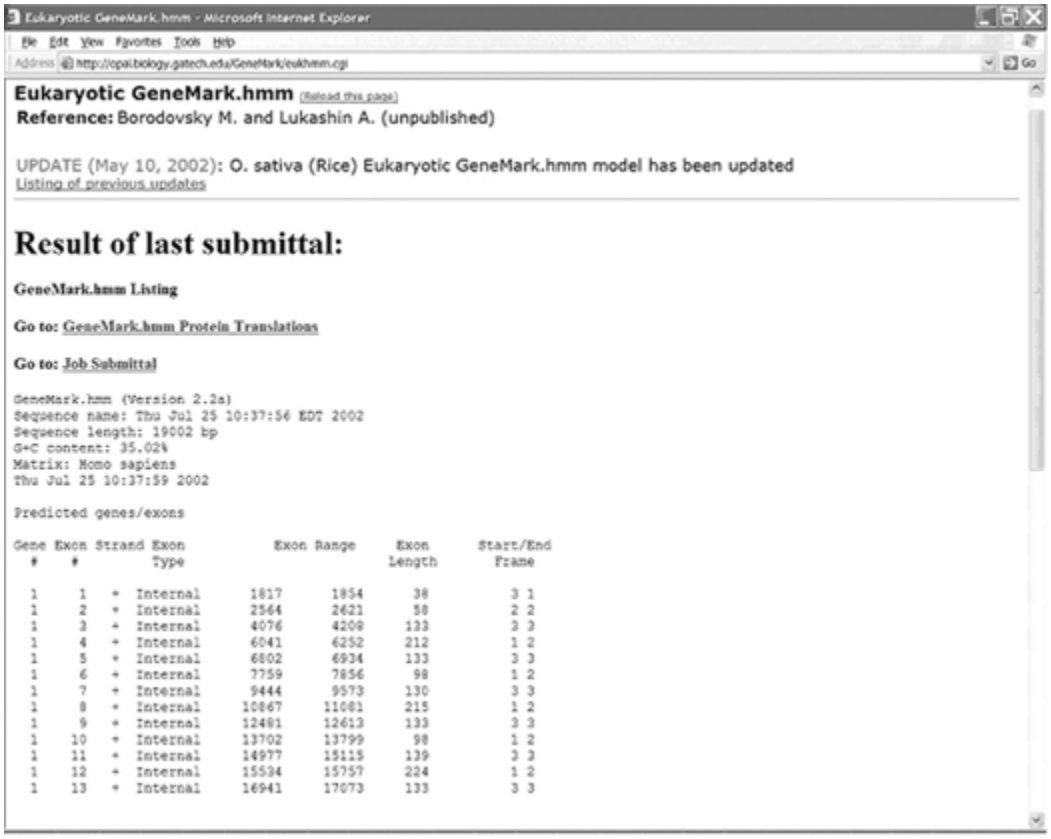

Eukaryotic GeneMark.hmm - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://opal.biology.gatech.edu/GeneMark/eukhmm.cgi
Go
Eukaryotic GeneMark.hmm (Reload this page)
Reference: Borodovsky M. and Lukashin A. (unpublished)
UPDATE (May 10, 2002): O. sativa (Rice) Eukaryotic GeneMark.hmm model has been updated
Listing of previous updates
Input Sequence
Title (optional):
Sequence:
>gi|178343|gb|M12523.1|HUMALBGC Human serum albumin (ALB)
gene, complete cds, 19002 bases, 807 checksum.
Sequence File upload:
Browse...
Species: H.sapiens
Output Options
Email Address: (required for graphical output or sequences longer than 100000 bp)
rivanov@amber.biology.gatech.e
Generate PostScript graphics
Print GeneMark 2.4 predictions in addition to GeneMark.hmm predictions
Translate predicted genes into protein
Run
Default
Start GeneMark.hmm

Figure 4.6.1 The user interface for the eukaryotic GeneMark.hmm program. Required input includes a DNA sequence in FASTA format, either copied and pasted into the Sequence text box, or uploaded from one's local drive using the Browse button next to the Sequence File Upload box.

downloaded as file `M12532.fna` at *http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm*. The sequence can be pasted into a sequence window if shorter than 100 kb or uploaded as a file if longer. The last exon of this gene, shown in bold in Figure 4.6.3, is alternatively spliced as either human serum albumin or human alloalbumin Venezia.

1. Via a Web browser, connect to *http://opal.biology.gatech.edu/GeneMark/eukhmm.cgi*. In the Input Sequence section, paste the DNA sequence into the Sequence box area and provide an optional sequence title in the Sequence Title box. (Fig. 4.6.1) If using the sample sequence (see above), copy and paste the human albumin DNA sequence (M12523) in FASTA format into the Sequence box or upload the file `M12523.fna` from a local drive using the Browse button next to the Sequence File Upload box.

   The Sequence File Upload option is the universal method for sending DNA sequences to the GeneMark.hmm server, as the copy and paste method has a limit on the length of the sequence. If the sequence has a FASTA (APPENDIX 1B) title line (e.g., `> Sequence name`), this name will be assigned to the sequence in the output, unless the user has given a name in the Sequence Title area. For the purpose of analysis, the FASTA title line, all numbers, and all white-space characters (e.g., spaces, tabs, and line returns) are ignored. In addition, all ambiguous letters and ASCII symbols other than the symbols of the four standard nucleotides, assuming that they occur rarely, are replaced with `C`. This minimizes the chance of the possible creation of a false start or stop codon.



Eukaryotic GeneMark.hmm - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address http://opal.biology.gatech.edu/GeneMark/eukhmm.cgi

Eukaryotic GeneMark.hmm (Reload this page)

Reference: Borodovsky M. and Lukashin A. (unpublished)

UPDATE (May 10, 2002): O. sativa (Rice) Eukaryotic GeneMark.hmm model has been updated

Listing of previous updates

Result of last submittal:

GeneMark.hmm Listing

Go to: GeneMark.hmm Protein Translations

Go to: Job Submittal

```
GeneMark.hmm (Version 2.2a)
Sequence name: Thu Jul 25 10:37:56 EDT 2002
Sequence length: 19002 bp
G+C content: 35.02%
Matrix: Homo sapiens
Thu Jul 25 10:37:59 2002

Predicted genes/exons
```

Gene #	Exon #	Strand	Exon Type	Exon Range		Exon Length	Start/End Frame
1	1	+	Internal	1817	1854	38	3 1
1	2	+	Internal	2564	2621	58	2 2
1	3	+	Internal	4076	4208	133	3 3
1	4	+	Internal	6041	6252	212	1 2
1	5	+	Internal	6802	6934	133	3 3
1	6	+	Internal	7759	7856	98	1 2
1	7	+	Internal	9444	9573	130	3 3
1	8	+	Internal	10867	11081	215	1 2
1	9	+	Internal	12481	12613	133	3 3
1	10	+	Internal	13702	13799	98	1 2
1	11	+	Internal	14977	15115	139	3 3
1	12	+	Internal	15534	15757	224	1 2
1	13	+	Internal	16941	17073	133	3 3

Figure 4.6.2 The text output for the Eukaryotic GeneMark.hmm program.

Albumin CDS:

```
CDS   join(1776..1854,2564..2621,4076..4208,6041..6252,
      6802..6934,7759..7856,9444..9573,10867..11081,
      12481..12613,13702..13799,14977..15115,15534..15757,
       16941..17073, 18526..18555)
```

Alloalbumin Venezia CDS:

```
CDS   join(1776..1854,2564..2621,4076..4208,6041..6252,
      6802..6934,7759..7856,9444..9573,10867..11081,
      12481..12613,13702..13799,14977..15115,15534..15757,
       16941..17073, 17688..17732)
```

Figure 4.6.3 The DNA sequence of complete human serum albumin (*ALB*) gene (GenBank accession no. M12523) was used as sample input for the program. The last exon of this gene (shown in bold) is alternatively spliced as either human serum albumin or human alloalbumin Venezia.

Large DNA sequences should be split into smaller ones that are more homogeneous in GC composition. For example, for H. sapiens, the recommended sequence size is ~100 kb. On the other hand, very short sequences should be avoided. To decrease the error rate in locating the initialization and termination sites, extended margins (1 to 2 kb) around the sequences of interest are recommended. Masking of DNA sequence repeats is not necessary.

2. Scroll down the page and select the name of the species of interest from the Species pull-down menu (Fig. 4.6.1). If using the sample sequence, select *H. sapiens* in that menu.

   The user has to make sure that the Species name, located below the Input Sequence section, is selected correctly. Choosing the right species name is essential to obtaining meaningful results, as the program automatically chooses the statistical model for the sequence analysis with regard to the given name. Currently (as of August, 2002), models are available for H. sapiens, C. elegans, A. thaliana, D. melanogaster, C. reinhardtii, Z. mays, T. aestivum, H. vulgare, M. musculus, and O. sativa.

   The strand of the DNA sequence does not have to be specified because prediction is performed on both DNA strands simultaneously.

3. Scroll further down the page and set the Output Options (Fig. 4.6.1). If using the sample sequence provided, enter a valid E-mail address and check all of the three check boxes: Generate PostScript Graphics, Print GeneMark 2.4 Predictions, and Translate Predicted Genes into Proteins.

   By default, the program generates a list of predicted exons for each predicted gene. The user has the option of choosing graphical output (by checking Generate PostScript Graphics), a report of GeneMark predictions (by checking Print GeneMark 2.4 Predictions...), or additional list of the protein translations of the predicted genes (by checking Translate Predicted Genes into Proteins). A valid E-mail address is required if a sequence is longer than 100 kb or if the graphical output is requested. The PostScript file will be returned to the user by E-mail.

4. After completing the above entries, click the Start GeneMark.hmm button to start running the program.

   For sequences shorter than 100 kb, the result will be displayed on the screen. If the user supplied an E-mail address and checked Generate PostScript Graphics, a PostScript file will be E-mailed to the user as well. For longer sequences the results can be only obtained via E-mail.

5. Interpret the text output.

   The eukaryotic GeneMark.hmm text output (Fig. 4.6.2) contains a list of `Predicted genes/exons` *for each predicted gene in terms of sequence coordinates. Both complete and partial genes are predicted by the program. Partial exons are not predicted.*

   For each gene identified by the gene number in the first column, there could be one or more exons listed on separate lines and identified by the exon number in the second column. In the third column, exons predicted on the direct strand are indicated with a "+" sign while those predicted on the reverse strand are labeled with a minus sign ("−"). For genes in which multiple exons were predicted, the `Exon Type` *column may contain* `Initial`, `Internal`, *or* `Terminal`*—terms describing the type of predicted exon. For single-exon genes,* `Exon Type` *is assigned as* `Single`*. The list of predicted exons shows start and end positions of each exon in the* `Exon range` *columns. The* `Start/End Frame` *columns specify the positions of the first and the last nucleotide of an exon in terms of codon position.*

   If the Translate Predicted Genes into Proteins box was checked (see step 3), the sequences of predicted proteins will be displayed, in FASTA format, in the output below the list of predicted genes and exons (Fig. 4.6.4).

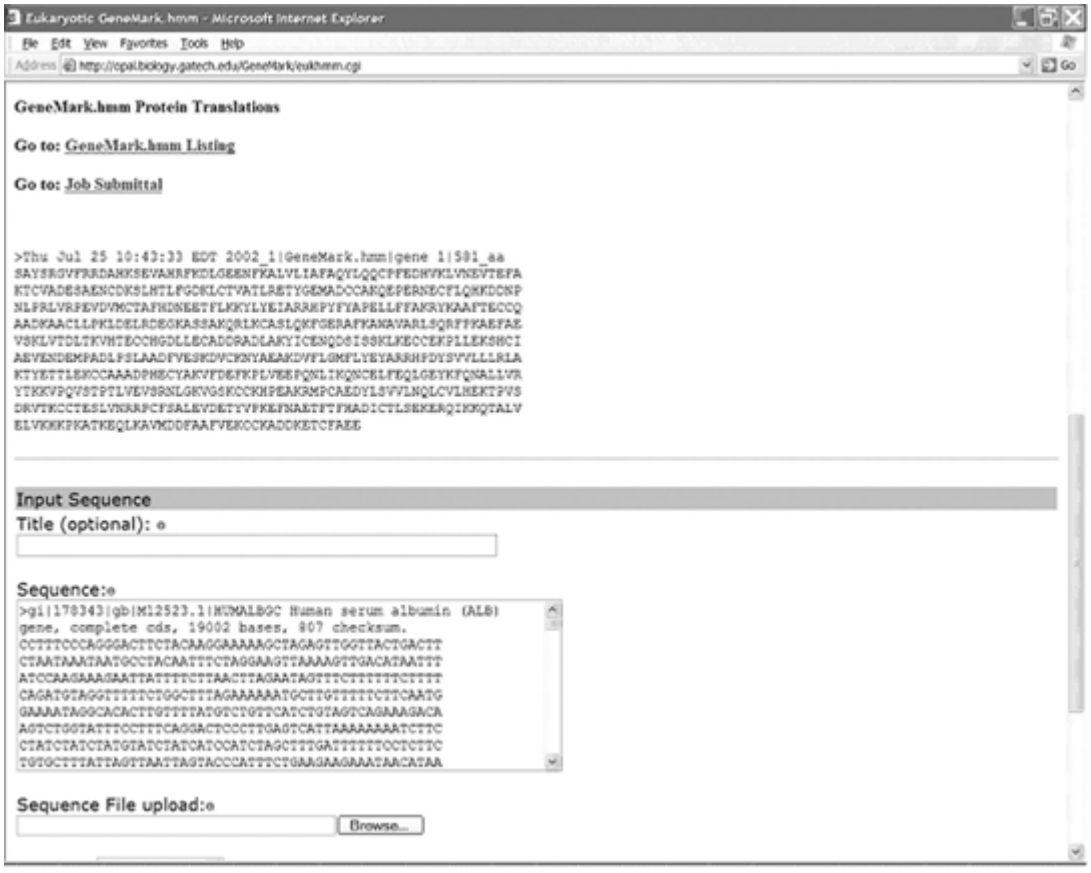


Figure 4.6.4 The text output the Eukaryotic GeneMark.hmm program containing the sequence of predicted protein in FASTA format.

Figure 4.6.2 displays the results for gene prediction using the sample sequence for human serum albumin (ALB) gene (M12523). Thirteen of fourteen annotated exons are predicted, and twelve of the predicted exons are predicted exactly, matching the start and the end of each exon. The predicted start position of the initial exon does not match the annotation and the last exon known to be alternatively spliced is not predicted. Prediction of initial and terminal exons is usually problematic in all gene-prediction programs due to the rather weak statistical patterns of the translation start and stop.

Figure 4.6.4 displays the predicted sequence of human serum albumin (ALB). The annotated sequences for the two proteins resulting from two variants of alternative splicing in the last exon are shown in Figure 4.6.5. The amino acid residues not included in the prediction are shown in bold in the annotated sequence.

6. Interpret the graphical output.

The GeneMark.hmm graphical output is always combined with the GeneMark graphical output (Fig. 4.6.6; also see UNIT 4.5, Basic Protocol). Graphical output in the PostScript format is generated by the GeneMark program using the models relevant to the species in question. The file is generated and E-mailed to the user if the Generate PostScript Graphics box was checked (see step 3). In this case GeneMark runs concurrently with the eukaryotic GeneMark.hmm program. While primarily used for prokaryotic DNA analysis (see UNIT 4.5), GeneMark can also be used to aid eukaryotic gene finding.

The GeneMark graphical output (Fig. 4.6.6) identifies regions with high coding potential, both inside and outside of open reading frames. These regions can be used together with the exon predictions of GeneMark.hmm to further analyze the sequence. Particularly, it

```
>gi|178344|gb|AAA98797.1| albumin| 609 aa
MKWVTFISLLFLFSSAYSRGVFRRDAHKSEVAHRFKDLGEENFKALVLIA
FAQYLQQCPFEDHVKLVNEVTEFAKTCVADESAENCDKSLHTLFGDKLCT
VATLRETYGEMADCCAKQEPERNECFLQHKDDNPNLPRLVRPEVDVMCTA
FHDNEETFLKKYLYEIARRHPYFYAPELLFFAKRYKAAFTECCQAADKAA
CLLPKLDELRDEGKASSAKQRLKCASLQKFGERAFKAWAVARLSQRFPKA
EFAEVSKLVTDLTKVHTECCHGDLLECADDRADLAKYICENQDSISSKLK
ECCEKPLLEKSHCIAEVENDEMPADLPSLAADFVESKDVCKNYAEAKDVF
LGMFLYEYARRHPDYSVVLLLRLAKTYETTLEKCCAAADPHECYAKVFDE
FKPLVEEPQNLIKQNCELFEQLGEYKFQNALLVRYTKKVPQVSTPTLVEV
SRNLGKVGSKCCKHPEAKRMPCAEDYLSVVLNQLCVLHEKTPVSDRVTKC
CTESLVNRRPCFSALEVDETYVPKEFNAETFTFHADICTLSEKERQIKKQ
TALVELVKHKPKATKEQLKAVMDDFAAFVEKCCKADDKETCFAEEGKKLV
AASQAALGL

>gi|178345|gb|AAA98798.1| alloalbumin Venezia|604 bases
MKWVTFISLLFLFSSAYSRGVFRRDAHKSEVAHRFKDLGEENFKALVLIA
FAQYLQQCPFEDHVKLVNEVTEFAKTCVADESAENCDKSLHTLFGDKLCT
VATLRETYGEMADCCAKQEPERNECFLQHKDDNPNLPRLVRPEVDVMCTA
FHDNEETFLKKYLYEIARRHPYFYAPELLFFAKRYKAAFTECCQAADKAA
CLLPKLDELRDEGKASSAKQRLKCASLQKFGERAFKAWAVARLSQRFPKA
EFAEVSKLVTDLTKVHTECCHGDLLECADDRADLAKYICENQDSISSKLK
ECCEKPLLEKSHCIAEVENDEMPADLPSLAADFVESKDVCKNYAEAKDVF
LGMFLYEYARRHPDYSVVLLLRLAKTYETTLEKCCAAADPHECYAKVFDE
FKPLVEEPQNLIKQNCELFEQLGEYKFQNALLVRYTKKVPQVSTPTLVEV
SRNLGKVGSKCCKHPEAKRMPCAEDYLSVVLNQLCVLHEKTPVSDRVTKC
CTESLVNRRPCFSALEVDETYVPKEFNAETFTFHADICTLSEKERQIKKQ
TALVELVKHKPKATKEQLKAVMDDFAAFVEKCCKADDKETCFAEEPTMRI
RERK
```

Figure 4.6.5 The annotated sequences for the two proteins resulting from two variants of alternative splicing in the last exon of human serum albumin are shown. The amino acid residues not included in the prediction (Fig. 4.6.4) are shown in bold.

may help to identify additional candidates for alternative splicing, which GeneMark.hmm alone would not detect, as it only predicts a single structure for each gene. The regions with high coding potential can be reported by GeneMark in text format as described in UNIT 4.5.

Figure 4.6.6 depicts the coding potential in the six possible reading frames (three frames on the direct strand and three frames on the reverse strand). An unbroken horizontal line at the 0.5 level indicates an open reading frame (ORF). Vertical ticks under the 0.5 level represent one of the three stop codons, TAA, TGA, or TAG. The thick gray horizontal line indicates a region with higher than expected coding potential predicted by GeneMark. The "<" and ">" marks represent putative exon boundaries, acceptor and donor splice sites, respectively, within a region of high coding potential. The thick black horizontal lines at the bottom level of each panel indicate exons predicted by GeneMark.hmm.

Eukaryotic GeneMark.hmm presents the graphical output in Adobe PostScript format, which is sent by E-mail to the user. Some E-mail readers will allow the figure to be displayed automatically, while others may require installation of additional software. In cases where the PostScript file was augmented with additional symbols due to the E-mail transfer, to extract a valid PostScript file, one needs to select every line (inclusive) between `%!PS-Adobe-2.0` *and* `%%EOF`*, and save it to a file with extension* `.ps`*. The file should not contain any blank lines before the* `%!PS-Adobe-2.0` *line or after the* `%%EOF` *line. This file can, then, be viewed with any PostScript viewer program, e.g., GhostView (available at http://www.cs.wisc.edu/~ghost/).*

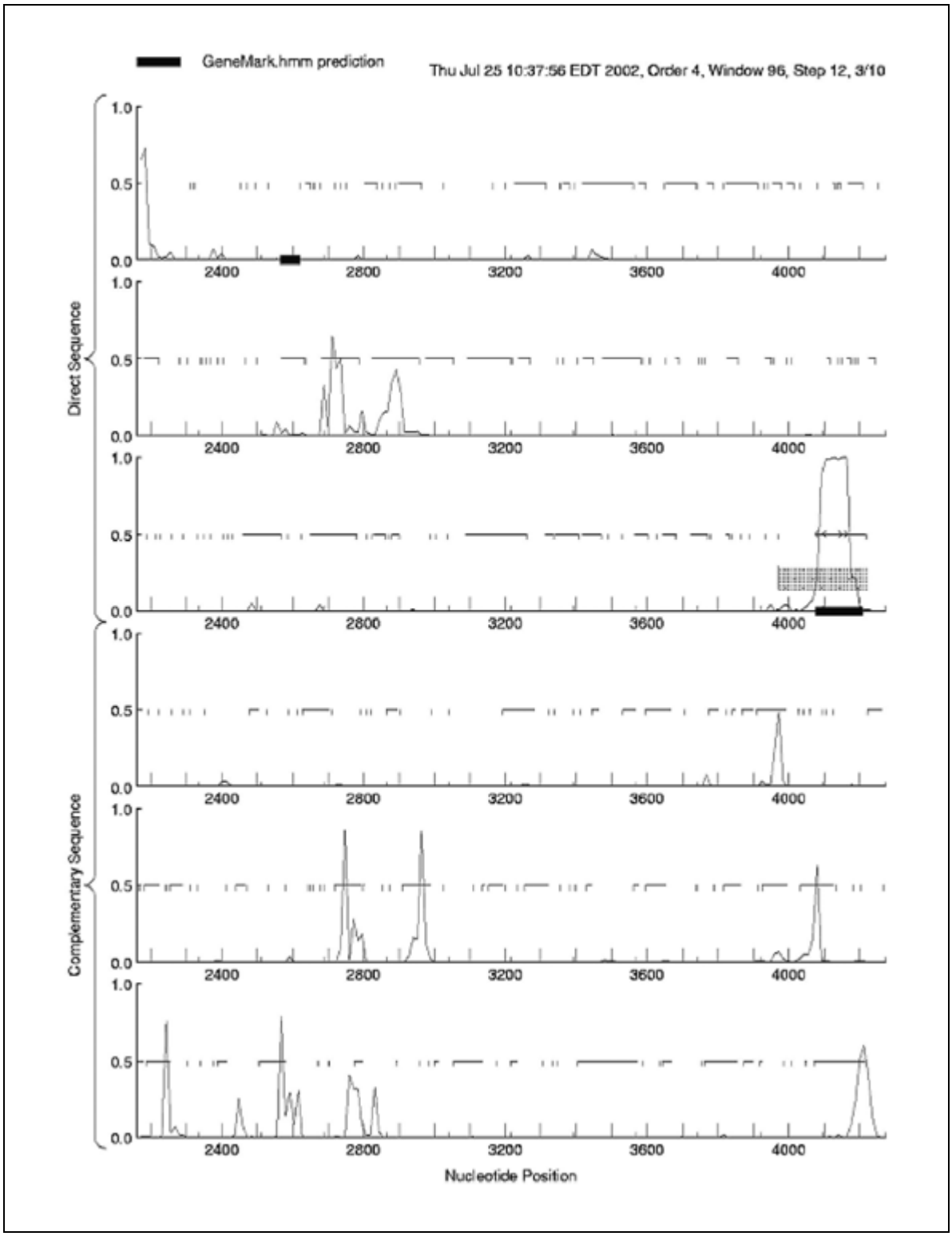


Figure 4.6.6 The graphical output from the Eukaryotic GeneMark.hmm program for a region of the example sequence. The six different panels represent the six possible reading frames, three each on the direct and reverse strands.

Figure 4.6.6 shows page 3 of the PostScript file generated by the GeneMark.hmm Web server for the sample file and viewed in the GhostView viewer. Two correctly predicted internal exons, 2564-2621 and 4076-4208, are shown as thick black lines. The first appears in the first frame of the direct strand and the second is in the third frame of the direct strand. High coding potential predicted by GeneMark is seen for the second exon (by a wide peak in the area of >4000 and by the thick gray line). High coding potential for the first exon cannot be seen due the small size (58 nt) of the exon, yet it is still correctly predicted by GeneMark.hmm.

ALTERNATE PROTOCOL 1

USING UNIX VERSION OF GeneMark.hmm

This protocol describes application of the stand-alone GeneMark.hmm for prediction of genes in eukaryotic genomes. An introduction to Unix is provided in *APPENDIX 1C*.

Necessary Resources

Hardware

Unix workstation with Linux, Sun Solaris, DEC Unix, SGI Irix, or IBM AIX operating system

Software

Stand alone Unix version of GeneMark.hmm, which is available through affiliated distributor (see information located at *http://opal.biology.gatech.edu/GeneMark/faq.html*)

Files

The model for a specific organism provided as a matrix file, e.g., `human.mtx`. The analyzed DNA sequence must be in FASTA format (*APPENDIX 1B*). The sample sequence—DNA sequence of complete human serum albumin (*ALB*) gene (GenBank accession no. M12523)—used to illustrate this protocol can be downloaded as file `M12532.fna` at *http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm*.

1a. From the Unix command line run GeneMark.hmm (program name `gmhmme`):

```
%gmhmme <DNA file> –m <matrix file> -o <output file>
```

The name of the compiled version of GeneMark.hmm is `gmhmme`. *The program requires at least two parameters: the name of the DNA sequence file and the name of a matrix file supplied after the* `-m` *option. The latter contains parameters of statistical models for DNA sequence analysis, generated from a training set of reliably annotated sequences from a particular organism. The species for which the matrix file was built must match the name of the species of the DNA sequence origin. By default, the output is saved in a file named after DNA sequence file with addition of the* `.lst` *extension. Option* `-o` *allows users to specify the output file name different from the default name. An example would be:*

```
%gmhmme m12523.fna –m human.mtx -o m12523.lst
```

1b. Alternatively, if GeneMark.hmm predictions are run routinely for the same organism, the matrix file name and the path to the file can be specified in the environmental variables `DEFMAT_HMME` and `MATPATH`, respectively. In the Unix csh and ksh shells, this can be done as shown in Figure 4.6.7. The matrix specifications can then be omitted on the command line. If using the sample sequence provided, the command will be as follows:

```
%gmhmme m12523.fna -o m12523.lst
```

The file `m12523.lst` will contain the output of the GeneMark.hmm program.

```
%setenv DEFMAT_HMME human.mtx                    (csh)
%setenv MATPATH /home/GeneMark.hmm/matrices      (csh)

>export DEFMAT_HMME=human.mtx                    (ksh)
>export MATPATH=/home/GeneMark.hmm/matrices      (ksh)
```

Figure 4.6.7 Specifying the matrix file name and path in the environmental variables `DEFMAT_HMME` and `MATPATH` in Unix and in ksh shells.

```
GeneMark.hmm (Version 2.2a)
Sequence name: HALB.gb.fna
Sequence length: 19002 bp
G+C content: 35.02%
Matrices file: ../../MTX/human0.mtx (Homo sapiens)
Thu Jul 25 16:04:48 2002

Predicted genes/exons

Gene Exon Strand Exon          Exon Range      Exon      Start/End
  #    #         Type                        Length      Frame
  1    1   +  Internal      1817      1854      38          3 1
  1    2   +  Internal      2564      2621      58          2 2
  1    3   +  Internal      4076      4208     133          3 3
  1    4   +  Internal      6041      6252     212          1 2
  1    5   +  Internal      6802      6934     133          3 3
  1    6   +  Internal      7759      7856      98          1 2
  1    7   +  Internal      9444      9573     130          3 3
  1    8   +  Internal     10867     11081     215          1 2
  1    9   +  Internal     12481     12613     133          3 3
  1   10   +  Internal     13702     13799      98          1 2
  1   11   +  Internal     14977     15115     139          3 3
  1   12   +  Internal     15534     15757     224          1 2
  1   13   +  Internal     16941     17073     133          3 3

Predicted gene sequence(s):

>HALB.gb.fna|GeneMark.hmm|gene 1|581_aa
SAYSRGVFRRDAHKSEVAHRFKDLGEENFKALVLIAFAQYLQQCPFEDHVKLVNEVTEFA
KTCVADESAENCDKSLHTLFGDKLCTVATLRETYGEMADCCAKQEPERNECFLQHKDDNP
NLPRLVRPEVDVMCTAFHDNEETFLKKYLYEIARRHPYFYAPELLFFAKRYKAAFTECCQ
AADKAACLLPKLDELRDEGKASSAKQRLKCASLQKFGERAFKAWAVARLSQRFPKAEFAE
VSKLVTDLTKVHTECCHGDLLECADDRADLAKYICENQDSISSKLKECCEKPLLEKSHCI
AEVENDEMPADLPSLAADFVESKDVCKNYAEAKDVFLGMFLYEYARRHPDYSVVLLLRLA
KTYETTLEKCCAAADPHECYAKVFDEFKPLVEEPQNLIKQNCELFEQLGEYKFQNALLVR
YTKKVPQVSTPTLVEVSRNLGKVGSKCCKHPEAKRMPCAEDYLSVVLNQLCVLHEKTPVS
DRVTKCCTESLVNRRPCFSALEVDETYVPKEFNAETFTFHADICTLSEKERQIKKQTALV
ELVKHKPKATKEQLKAVMDDFAAFVEKCCKADDKETCFAEE
```

Figure 4.6.8 The text output for the Unix version of Eukaryotic GeneMark.hmm program.

2. Interpret the results.

   As in the Web-interface version, the first lines of the output file (Fig. 4.6.8) contain a description of the parameters for GeneMark.hmm such as version, sequence file name, sequence length, GC content, matrix file name, and time and date of prediction. Following this information there is a table of predicted genes and exons in the format shown in Figure 4.6.2 (see Basic Protocol, step 5, for detailed explanation). The predicted protein sequences in FASTA format are listed in the end of the output file.

   As expected, the output file for Unix version of GeneMark.hmm contains the same predictions for complete human serum albumin (ALB) gene as the Web-interface version (see Basic Protocol).

ALTERNATE PROTOCOL 2

USING GeneMarkS FOR GENE FINDING IN EUKARYOTIC VIRUSES

This protocol is used for analyzing eukaryotic viruses with a modified version of GeneMarkS, described in Alternate Protocol 2 of *UNIT 4.5* (Fig. 4.5.8). GeneMarkS can be accessed through the Web at (*http://opal.biology.gatech.edu/GeneMark/genemarks.cgi*). This program can analyze an anonymous sequence of a eukaryotic virus, derive necessary statistical models, and predict genes. The major difference between this program and the prokaryotic version of GeneMarkS is that, for sufficiently long genomes, instead of deriving a model of the ribosomal binding site, the eukaryotic version of GeneMarkS derives a Kozak-like pattern near the gene start. For sequences shorter than 100 kb, both versions operate in essentially the same way, using heuristic models. As described in step 3 below, these models are then used in GeneMark.hmm to predict protein coding regions.

Necessary Resources

Hardware

A personal computer or workstation with Web access

Software

A Web browser e.g., Netscape Communicator or Microsoft Internet Explorer

Files

A single sequence in FASTA format (*APPENDIX 1B*). See Alternate Protocol 2 in *UNIT 4.5* for the example sequence used in Figure 4.5.8.

1. Via a Web browser, connect to *http://opal.biology.gatech.edu/GeneMark/genemarks.cgi*. In the Input Sequence section, paste an input sequence into the Sequence box area, or, alternatively, click on Browse next to the Sequence File Upload box to upload the input sequence file from a local drive.

   The Sequence File Upload option is more powerful, since the copy and paste method imposes a limit on the length of the sequence. If the sequence has a FASTA (APPENDIX 1B) title line (e.g., `> Sequence name`), this name will be assigned to the sequence in the output, unless the user gave a name in the Sequence Title text area. For the purpose of analysis, all nonalphabet symbols are ignored and all ambiguous letters other than the symbols of the four nucleotides, assuming that they occur rarely, are replaced with `C`.

2. Scroll down the page to the Running Options and select the option Use Eukaryotic Virus Version.

3. Scroll further down the page and set the Output Options. The user may request the graphical output. An E-mail address is required for sending text output, for sequences longer than 1 Mbp, or if graphical output is requested.

4. After completing the above entries, click the Start GeneMarkS button. The results will be depicted on the browser or will be sent to the E-mail address provided.

5. Interpret the text output.

   The text output from GeneMarkS is identical to that of GeneMark.hmm (see UNIT 4.5, Alternate Protocol 1).

6. Interpret the graphical output.

   The graphical output from GeneMarkS is identical to that of GeneMark.hmm (see UNIT 4.5, Alternate Protocol 1) when using just one model.

GUIDELINES FOR UNDERSTANDING RESULTS

The accuracy of prediction is described in terms of sensitivity and specificity, both defined for each signal site (translational start and stop, donor, and acceptor) and for each exon as a whole. Specificity is defined as the number of true predicted sites (exons) over the number of all predicted sites (exons). This measure characterizes how many sites are overpredicted, e.g., the higher the specificity, the lower the overprediction rate, and the more reliable the predictions. On the other hand, sensitivity is defined as the number of true predicted sites (exons) over the whole number of annotated sites (exons). The higher the sensitivity, the lower the chance that a true site (exon) has not been predicted. This measure characterizes how well the program recognizes the actual sites or exons. The two measures of accuracy—specificity and sensitivity—are inversely related, with one increasing when the other decreases. The accuracy of two programs may be considered similar if the increase (decrease) of specificity is "compensated" by the decrease (increase) in the sensitivity. Therefore, when comparing the performance of different gene-prediction programs it is important to take into account both measures of accuracy.

GeneMark.hmm produces sufficiently accurate prediction results with high sensitivity and specificity for a wide range of eukaryotic organisms. These results are achieved by tuning up the organism-specific models (see Background Information, discussion of Model Construction).

COMMENTARY

Background Information

The GeneMark.hmm algorithm (Lukashin and Borodovsky, 1998; M. Borodovsky and A.V. Lukashin, unpub. observ.) was designed to improve gene-prediction quality in terms of finding exact exon/intron and gene boundaries. The previously developed program GeneMark (Borodovsky and McIninch, 1993) identified a gene mainly as the open reading frame where the gene resides. However, the Web site version of the GeneMark program does not use a notion of exons and introns. Therefore, it does not predict exon/intron boundaries as such. The underlying idea of GeneMark.hmm was to embed the GeneMark models for protein-coding (exons) and non-coding (intron and intergenic) regions into a naturally derived hidden Markov model (HMM) framework with exon/intron boundaries modeled as transitions between hidden states.

Hidden Markov model framework

The HMM framework of GeneMark.hmm, the logic of transitions between hidden Markov states, followed the logic of the genetic structure of eukaryotic genome. The Markov models of coding and non-coding regions were incorporated into the HMM framework to generate stretches of DNA sequence with coding or noncoding statistical patterns. This type of HMM architecture is known as "HMM with duration" (Rabiner, 1989). The sequence of hidden states associated with a given DNA sequence carries information on positions where coding function is switching into noncoding and vice versa. The sequence of hidden states constitutes the HMM trajectory. The core GeneMark.hmm procedure, the dynamic programming type algorithm (Rabiner, 1989), finds the most likely HMM trajectory given the DNA sequence. To further improve the predic-

tion of the exons, the models of the translation start, translation end, acceptor, and donor were derived. At the post-processing stage, these models are used to refine translation initiation and termination codon predictions as well as the predictions of exon/intron boundaries. However, the range of uncertainty for the initiation and termination codon positions still presents a problem for eukaryotic GeneMark.hmm, as well as for a majority of other gene-prediction programs, due to relatively weak statistical patterns for these sites.

Model construction

The goal of model construction is to maximize both the specificity and the sensitivity of gene predictions. For construction of the model, the sequences specific to a given organism and having a reliable sequence annotation are selected for the training set. They are verified for sequencing irregularities and further clustered by their GC contents. Each cluster must contain at least 1.2 Mb to achieve reasonable accuracy. The Markov models of several orders are generated for coding and noncoding regions. From the same training set of sequences, the position frequency matrices are constructed for translational start and termination sites as well as for donor and acceptor splice sites. Other parameters, including length distribution for introns and exons, are also generated from the training set. All parameters of the model are concatenated into a single matrix file and encrypted in order to avoid unintentional modification of crucial parameters.

To date, GeneMark.hmm models have been constructed for several eukaryotic organisms, covering important classes of low and high eukaryotes such as invertebrates (*C. elegans*), green algae (*C. reinhardtii*), insects (*D. melanogaster*), plants (*Z. mays, O. sativa, A. thaliana, T. aestivum, H. vulgare*), and mammals (*H. sapiens, M. musculus).*

Critical Parameters and Troubleshooting

GeneMark.hmm has been thoroughly tested since 1997. Yet, there are some sequences that can cause termination of the program without output. If a user encounters this problem the authors would appreciate it if the user sends the sequence and the name of the matrix used to the administrator of the GeneMark.hmm Web page at the E-mail link on the page. Organism-specific models cannot be edited by users. For time efficiency of calculation, values of several variables are restricted, e.g., the maximum intron length is set to 30 kb. Some genes, i.e., those with introns larger than this limit, will not be predicted correctly since the program will be forced to predict artificial exons in the intron region larger than 30 kb.

Suggestions for Further Analysis

The protein translations of the predicted genes can be further analyzed by BLASTP (Altschul et al., 1990; *UNIT 3.3*) to make functional prediction for the putative genes. Experimental biologists can use the sequences around the predicted exons and genes to create primers for PCR analysis of genes of interest as well as to design oligonucleotides representing protein-coding regions for DNA expression arrays.

Literature Cited

Altschul, S.F., Gish, W., et al. 1990. Basic local alignment search tool. *J. Mol. Biol.* 215:403-410.

Besemer, J., Lomsadze, A., and Borodovsky, M. 2001. GeneMarkS: A self-training method for prediction of gene starts in microbial genomes: Implications for finding sequence motifs in regulatory regions. *Nucleic Acids Res.* 29:2607-2618.

Borodovsky, M. and McIninch, J. 1993. GeneMark: Parallel gene recognition for both DNA strands. *Comput. Chem.* 17:123-133.

Lukashin, A.V. and Borodovsky, M. 1998. GeneMark.hmm: New solutions for gene finding. *Nucleic Acids Res.* 26:1107-1115.

Pearson, W.R. 1990. Rapid and sensitive sequence comparison with FASTP and FASTA. *Methods Enzymol.* 183:63-98.

Rabiner, L.R. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the I.E.E.E.* 77:257-286.

Internet Resources

http://opal.biology.gatech.edu/GeneMark

GeneMark.hmm Web site

http://opal.biology.gatech.edu/GeneMark/genemarks.cgi

GeneMarkS Web site

Contributed by Mark Borodovsky
School of Biology and School of Biomedical Engineering
Georgia Institute of Technology
Atlanta, Georgia

Alex Lomsadze, Nikolai Ivanov, and Ryan Mills
School of Biology
Georgia Institute of Technology
Atlanta, Georgia

Application of FirstEF to Find Promoters and First Exons in the Human Genome

Mammalian genomes contain vast amounts of *cis*-regulatory regions responsible for differential regulation of thousands of protein-coding genes. Identification of these regulatory regions, generally located upstream of the first exon, is a very important part of gene finding. First Exon Finder (FirstEF; Davuluri et al., 2001) was developed to predict first exons and promoters in the human genome. The FirstEF algorithm is a decision tree that consists of a set of quadratic discriminant functions (*UNIT 4.2*) at its nodes. The discriminant functions are optimized to find potential first donor sites and CpG-related and non-CpG-related promoter regions. For every potential first donor site (GT) and upstream promoter region, FirstEF decides whether or not the intermediate region can be a potential first exon, based on a set of quadratic discriminant functions.

An explanation of both the Web-based (see Basic Protocol) and local (see Alternate Protocol) versions of FirstEF to find potential promoters and first exons in human DNA are given. In addition, a discussion of how the user can combine the predictions of FirstEF with other information, such as mRNA/EST alignments or predictions of gene finding programs (e.g., GENSCAN; MZEF, *UNIT 4.2*), to create more reliable annotations is also presented (see Support Protocol).

USING WEB-BASED FirstEF TO PREDICT PROMOTERS AND FIRST EXONS

BASIC PROTOCOL

The user can submit a FASTA-formatted DNA sequence (*APPENDIX 1B*) to FirstEF either through the World Wide Web (*http://rulai.cshl.org/tools/FirstEF*) as described in this protocol, or through a locally installed version (see Alternate Protocol). Since the Web server has a restriction on the size of the input file, with the maximum being 100 kb, it is advisable to obtain a local copy of the FirstEF software for analyzing large genomic sequences (e.g., the entire DNA sequence of a human chromosome).

Necessary Resources

Hardware

Computer with Internet access (e.g., PC running Microsoft Windows or Linux, Apple Macintosh, Unix workstation)

Software

Internet browser (e.g., Netscape Navigator, Microsoft Internet Explorer)

Files

DNA sequences to be analyzed in FASTA format (*APPENDIX 1B*)

Sample sequences can be found at the FirstEF Web site. The example (Example 1) used in this unit is a DNA sequence of length 100 kb from human chromosome 20 (chr20:300001-400000, NCBI build 30), which can be found at the Current Protocols Web site (http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm).

Submit sequence to the FirstEF Web server

1. Access the FirstEF Web server through an Internet browser (*http://rulai.cshl.org/tools/FirstEF*). Upload the file containing the DNA sequence in FASTA format (maximum size 100 kb) or cut and paste the sequence into the sequence window.

A screen shot of the FirstEF Web page is given in Figure 4.7.1.

Contributed by Ramana V. Davuluri
Current Protocols in Bioinformatics (2003) 4.7.1-4.7.10
Copyright © 2003 by John Wiley & Sons, Inc.

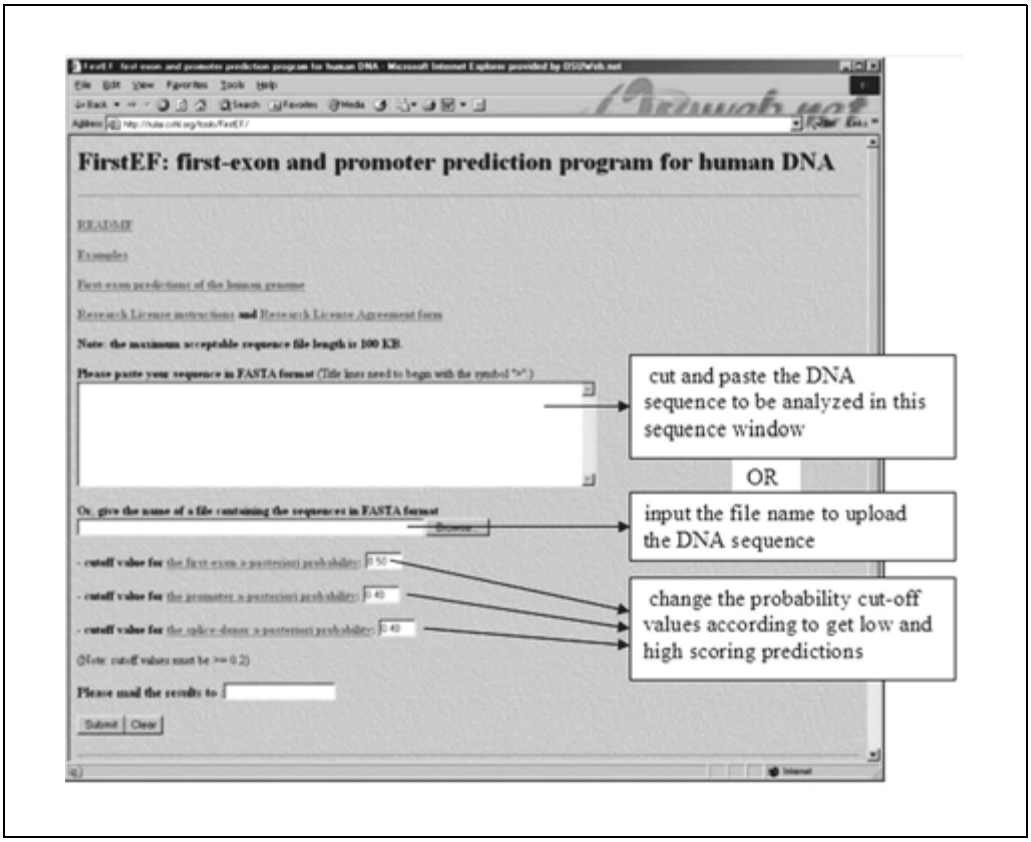

FirstEF: first-exon and promoter prediction program for human DNA
README
Examples
cut and paste the DNA sequence to be analyzed in this sequence window
OR
input the file name to upload the DNA sequence
change the probability cut-off values according to get low and high scoring predictions

Figure 4.7.1 The screen shot of the FirstEF Web page (see Internet Resources). The README file, available with the downloaded version, gives a brief description of how to use FirstEF. Examples of human genomic sequences analyzed by FirstEF are presented at the Web site. Preliminary annotations of the human genome can also be accessed from this Web site.

2. Enter desired values for the three parameters listed:

   a. The first-exon a-posteriori probability, P(exon).

      This value quantifies the probability of finding a true first exon at the predicted location. A value of P(exon) = 1 means that the first-exon prediction is 100% correct, whereas a value of P(exon) = 0 means that the first-exon prediction is 100% incorrect.

   b. The splice-donor a-posteriori probability, P(donor).

      This value quantifies the probability of finding a true splice-donor at the predicted location. A value of P(donor) = 1 means that the splice-donor prediction is 100% correct, whereas a value of P(donor) = 0 means that the splice-donor prediction is 100% incorrect.

   c. The promoter a-posteriori probability, P(promoter).

      This value quantifies the probability of finding a true promoter at the predicted location. A value of P(promoter) = 1 means that the promoter prediction is 100% correct, whereas a value of P(promoter) = 0 means that the promoter prediction is 100% incorrect.

      These user-selected values are the lower boundaries to the probabilities that are computed by the quadratic discriminant functions in the FirstEF algorithm: Exon-QDF, Promoter-QDF, and Donor-QDF (described in Davuluri et al., 2001). By default, FirstEF prints out the predictions of all first exons that satisfy the three constraints: P(exon) >0.5, P(donor) >0.4, and P(promoter) >0.4. This choice of cut-off values (0.5, 0.4, 0.4) results in a sensitivity and specificity of ~80%, based on cross validation analysis of real first exons (Davuluri et al., 2001).

Advanced users may wish to run FirstEF with different cut-off values in order to obtain more sensitive or more specific first-exon predictions. In principle, the user can adjust all three cut-off values independently of each other, which might be of interest in special circumstances where, for example, first exons with strong splice-donor sites—i.e., P(donor) ≥ 0.8—and weak promoters—i.e., P(promoter) ≤0.5—or weak splice-donor sites—i.e., P(donor) ≤ 0.5—and strong promoters—i.e., P(promoter) ≥ 0.8—are searched.

As a rule of thumb, it is recommend to modify only one parameter—e.g., P(exon)—and choose the other two cut-off values proportional to this value—e.g., P(donor) = 0.8 × P(exon) and P(promoter) = 0.8 × P(exon). Note that cut-off values below 0.2 are not accepted. A value of 0.8 was chosen to maintain balance between the parameters and has no special significance in and of itself.

3. Enter the E-mail address to which the results are to be sent (*optional*).

   If an E-mail address is provided, the server will forward the results in the body of an E-mail message.

4. Click Submit.

   The server displays the results in the browser.

Output of FirstEF

5. Analyze the resulting output (Fig. 4.7.2), which consists of eight columns (described in Table 4.7.1) and presented in two parts (predictions for direct and complementary strands) for boundaries of the promoter and first exon from the `Promoter` and `Exon` columns, respectively.

   For Example 1, FirstEF predicted five clusters on the direct strand and four clusters on the complementary strand. The predictions within a cluster are considered as probable alternative first exons of the same gene and ranked according to a posteriori probabilities. The cluster numbers of the prediction are displayed in the first column (`No.`*) and ranks in the last column (*`Rank`*). The prediction with rank* `1` *in cluster number* `1` *has a promoter spanning from* `16397` *to* `16966` *(of length 570 nt) and exon spanning from* `16897` *to* `17093`*. Note that the first 500 nucleotides of* `16397` *to* `16966` *is upstream of the region of the predicted TSS and the last 70 nucleotides are the downstream of the predicted TSS that overlaps with the predicted first exon (i.e.,* `16897` *to* `17093`*). Further, this particular prediction is CpG related, and hence the predicted CpG window* `15748` *to* `15949`*, of length 201, is reported in CpG window column.*

   Note that all promoters listed in this example have a length of 570 nt because the promoter QDF was trained on a promoter of length 570 nt.

6. Examine the a posteriori probabilities of promoter, exon, and donor, which are labeled `P(promoter)`, `P(exon)`, and `P(donor)`, respectively.

   The boundaries of `Exon` *are the transcription start sight (left boundary) and the donor site (right boundary).*

   The higher the values of the probabilities, the higher the chance that the corresponding predictions are real. A value of P(exon) = 1 means that the first-exon prediction is 100% correct, whereas a value of P(exon) = 0 means that the first exon prediction is 100% incorrect. P(exon) values ≥0.5 are considered significant.

7. Determine if the predicted first exon is CpG related.

   If the predicted first exon is CpG related, the boundaries of the corresponding `CpG Window` *of length 201 are presented. Otherwise, the* `CpG Window` *entry will read* `Non-CpG-related`*. See Background Information for further discussion of CpG windows.*

8. Observe the `Rank` of each first exon prediction within each cluster.

9. Combine the resulting predictions with other annotations (see Support Protocol).

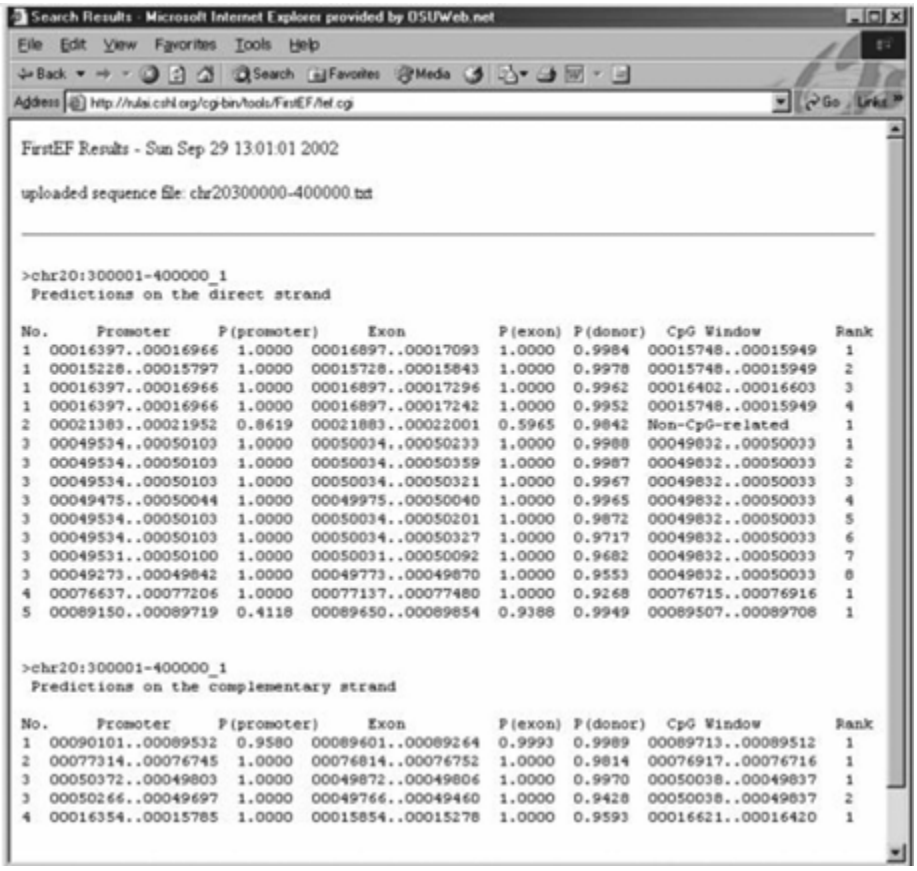


Figure 4.7.2 Screen shot of FirstEF output of Example 1 with default cut-off values for `P(exon)`, `P(promoter)`, and `P(donor)`. For each sequence in the input file, FirstEF presents predictions on direct and complementary strands separately. Sequence header follows the symbol `>` for each block of predictions. The line immediately following gives the strand information for the predictions which follow. Descriptions of each column are given in Table 4.7.1.

Table 4.7.1 Description of the Columns in FirstEF Output[a]

Column	Description
`No.`	Serial number of the predicted first exon cluster[b]
`Promoter`	Predicted promoter of length 570 bp
`P(promoter)`	A posteriori probability of promoter for a given window of size 570 bp
`Exon`	Predicted exon boundaries
`P(exon)`	A posteriori probability of exon for a given GT and promoter region
`P(donor)`	A posteriori probability of donor for a given GT
`CpG Window`	Boundaries of the CpG window of length 201, if the exon is CpG-related, otherwise the output reads `Non-CpG-related`
`Rank`	Rank of the first exon within a cluster

[a]Figure 4.7.2 shows output from Example 1.

[b]If two predicted first exons are separated by <1000 bp, they are considered as first-exon predictions of the same gene and ranked based on a posterior probabilities.

USING LOCAL FirstEF TO PREDICT PROMOTERS AND FIRST EXONS

ALTERNATE PROTOCOL

FirstEF can be obtained through the Office of Technology Transfer, Cold Spring Harbor Laboratory, by logging on to the FirstEF Web site, clicking the Research Licenses Instructions link, and following the directions provided. FirstEF software is freely available to nonprofit research institutions in executable form for Unix and Linux platforms. The software package includes a README file with instructions for installing the software. Note that a research license agreement will need to be signed.

Necessary Resources

Hardware

Computer workstation with Unix or Linux operating system

Software

FirstEF software (*http://rulai.cshl.org/tools/FirstEF*)

Files

DNA sequences to be analyzed in FASTA format (*APPENDIX 1B*)

Sample sequences can be found at the FirstEF Web site. The example (Example 1) used in this unit is a DNA sequence of length 100 kb from human chromosome 20 (chr20:300001-400000, NCBI build 30), which can be found at the Current Protocols Web site (http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm).

1. Type the following command at the Unix/Linux command prompt:

   ```
   firstef <input file name> <output file name>
   ```

   For example, if sequences to be analyzed are stored in a file called `example.seq`, type the following command to run FirstEF:

   ```
   firstef example.seq example.result
   ```

   Here the result file is named as `example.result`. *The user can choose the name of both the input and output files.*

2. Open the output file using a any text editor (*APPENDIX 1C*).

3. Analyze results as described (see Basic Protocol, steps 5 to 9).

   The main difference between the Web-based and local version of FirstEF is the size of the input file. The maximum acceptable size of the input file for the Web-based version is 100 kb, whereas there is no size limit for the local version. In fact, local FirstEF can analyze any human chromosome sequence all at once. However, in this version, the user has no option to select different cut-off values. Instead, the local version of FirstEF outputs all predictions with P(donor) ≥ 0.3, P(promoter) ≥ 0.3, and P(exon) ≥ 0.5. (Please check specific downloaded versions for these cut-off values.) The user must then manually parse these predictions for the cut-off values of their choice.

COMBINING FirstEF PREDICTIONS WITH OTHER ANNOTATIONS

SUPPORT PROTOCOL

As there is a wealth of data in sequence databases (GenBank/EMBL/DDBJ/ENSEMBL), such as mRNAs, ESTs, and other genomic annotations, the user can combine that information with FirstEF predictions to produce more reliable first exon and promoter annotations for the human genome. The procedure outlined in this protocol uses BLAST (*UNITS 3.3 & 3.4*) or MEGABLAST (*UNIT 3.3*), SIM4 (Florea et al., 1998), and a gene prediction program—either GENSCAN, MZEF (*UNIT 4.2*), or both—to find probable internal exons, and then combine them with FirstEF predictions.

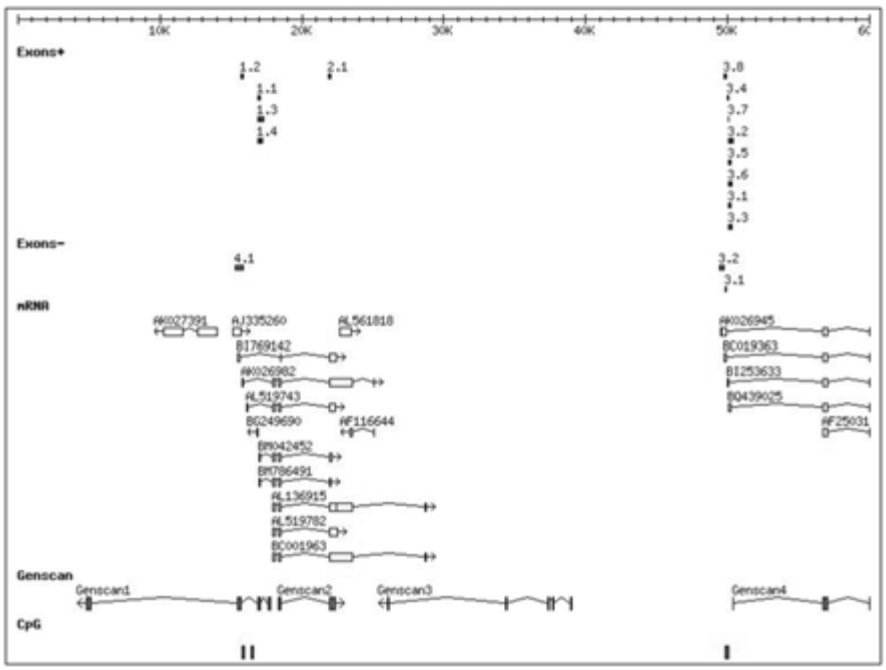


Figure 4.7.3 Part of the annotations for Example 1 (from 1 to 60000 bp) obtained by following the steps described (see Support Protocol). `Exons+` and `Exons-` indicate the predicted first exons on direct and complementary strands, respectively. The exon with symbol i.j represents j^{th}-ranked first exon in the i^{th} cluster. `AK027391` represents the transcript mapped to the exons complement (14015..12633, 11583..10224). Refer to Table 4.7.2 for this and the exon coordinates of other mapped cDNA/ESTs. For the coordinates of the first exons and CpG windows refer to Figure 4.7.2.

Necessary Resources

Hardware

Computer with Internet access (e.g., PC running Microsoft Windows or Linux, Apple Macintosh, or Unix workstation).

Software

Internet browser (e.g., Netscape Navigator, Microsoft Internet Explorer)
BLAST/MEGABLAST (*http://www.ncbi.nih.gov/BLAST*; *UNITS 3.3 & 3.4*)
SIM4 (*http://pbil.univ-lyon1.fr/sim4.html*)
GENSCAN (*http://genes.mit.edu/GENSCAN.html*)
MZEF (*http://www.cshl.edu/mzhanglab*; *UNIT 4.2*)

Files

The DNA sequence of interest in FASTA format (*APPENDIX 1B*)

1. BLAST the sequence against the nr and EST databases using BLASTN (*UNIT 3.3*) or MEGABLAST (*UNIT 3.4*) in case of very long sequences. Note the list of accession numbers from the BLAST output of cDNAs and ESTs with a percent identity score ≥99.

2. Use SIM4 (*http://pbil.univ-lyon1.fr/sim4.html*; Florea et al., 1998) to align each of the cDNA/ESTs with the genomic sequence so as to identify exons with canonical splice sites.

Table 4.7.2 Annotations and Corresponding Exon Coordinates for Example 1[a]

Annotation	Mapped/predicted exon boundaries
AK027391	14015..12633, 11583..10224
AJ335260	15142..15732
BI769142	15443..15615, 18512..18533, 21911..22419
AK026982	15814..15843, 17950..18064, 18339..18533, 21911..23573, 25046..25058
AL519743	16138..16213, 17950..18064, 18339..18533, 21911..22314
BG249690	Complement 16923..16799
BM042452	17001..17093, 17950..18064, 18339..18533, 21911..22135
BM786491	17007..17093, 17950..18064, 18339..18533, 21911..22065
AL136915	17948..18064, 18339..18533, 21911..22417, 22432..23573, 28702..28718
AL519782	17951..18064, 18339..18533, 21911..22530
BC001963	17953..18064, 18339..18533, 21911..23565, 28702..28718
GENSCAN2	18346..18531, 21909..22334
AL561818	22653..23474
AF116644	Complement 25060..25044, 23565..23352
GENSCAN3	Complement 39001..38950, 37758..37662, 37426..37330, 34439..34339, 26114..26009
AK026945	49544..49585, 49619..49870, 56712..57002, 59988..60280, 64899..66260, 67239..67251
BC019363	49764..49870, 56712..57002, 59988..60280, 64899..66260, 70389..70417, 71182..71216
BI253633	50014..50092, 56712..57002, 59988..60280, 64899..65000
BQ439025	50059..50201, 56712..57002, 59988..60281
AF250311	56712..57002, 59988..60286, 64899..65391
GENSCAN4	50358..50360, 56713..57003, 59989..60281, 64900..65392

[a]The graphical depiction is presented in Figure 4.7.3.

3. Submit the sequence to GENSCAN (*http://genes.mit.edu/GENSCAN.html*) and/or MZEF (*http://www.cshl.edu/mzhanglab*; *UNIT 4.2*) gene prediction programs, and select the consensus predictions (exons).

4. Merge the exons belonging to overlapping cDNAs/ESTs into a single transcript. If there are no overlapping cDNA/ESTs to support GENSCAN/MZEF predictions, consider them as probable novel exons. If the cDNA/EST alignments have different splice patterns, keep them separate.

5. Merge each of the assembled transcripts with a FirstEF predicted cluster that falls on the same strand. If a predicted first exon overlaps with the first exon of assembled transcript from step 4, then consider that as the first exon of the corresponding transcript. Otherwise, assign the nearest 5′ first exon prediction to the assembled transcript.

Figure 4.7.3 shows the assembled transcripts and first exon predictions for the DNA sequence that spans 1 to 60000 bp in Example 1. Table 4.7.2 presents the annotations and their corresponding exon coordinates.

GUIDELINES FOR UNDERSTANDING RESULTS

Like any gene prediction program based on pattern recognition methods, FirstEF has false positive (i.e., a genomic region that is not a real exon is predicted as real exon) and false negative (i.e., a genomic region that is in fact a real exon is not predicted by FirstEF) predictions. The user should experiment with different cut-off probability values and consider three important points in interpreting the results of FirstEF.

CpG Related and Non-CpG-Related First Exons

The accuracy of FirstEF is higher for CpG-related first exons, with an average sensitivity and specificity above 90%. Hence, the predictions with CpG window and probability values higher than 0.9 are very likely to be real. In case of non-CpG-related first exons, the accuracy of FirstEF is relatively low, with average sensitivity and specificity just above 70%. In other words, FirstEF may miss roughly 3 out of 10 real non-CpG-related first exons and 3 out of 10 non-CpG-related first exons may be false predictions that are actually not real.

First Exon Clusters

If two first exon predictions are separated by <1000 bp, FirstEF considers them as probable first exons of the same gene and places them in the same cluster. If a cluster has more than three predictions, at least one of those in that cluster is highly likely to be a real first exon. Other predictions in the cluster may be alternative first exons, which may need additional support such as a cDNA/EST match.

Predictions on the Alternative Strand

FirstEF predicts first exons on both positive and negative strands. If there is a strong promoter region, particularly in CpG islands, FirstEF tends to predict an overlapping first exon on the opposite strand due to a strong donor site. In such cases, the predictions that have downstream cDNA/EST match should be accepted. If there is no supporting cDNA/EST match on both sides of the predicted cluster, then the one that has the higher probability values should be considered. The predictions of FirstEF on the alternative strand that fall inside a gene transcript (assembled transcript of cDNA/EST) are potential false positives and should be ignored.

COMMENTARY

Background Information

First exons, promoters and CpG windows

Gene finding is one of the most vital phases of genome annotation. Sequence homology is perhaps the most important evidence used to detect functional elements in genomic sequences. A direct comparison of a genomic sequence (Example 1) with cDNA/ESTs can identify regions of the query sequence that correspond to transcribed genes. However, most of the ESTs and cDNA sequences are 5′ incomplete and do not provide information about the first exons and promoter regions. On the other hand, gene finding programs, such as GENSCAN (Burge and Karlin, 1997) and MZEF (Zhang, 1997; *UNIT 4.2*) were trained to predict protein coding exons only. Detecting the gene regulatory regions is important not only to annotate the genome, but also help to understand the large-scale gene expression data, such as those from microarray experiments (see Chapter 7). FirstEF is the only program that was specifically designed to predict both partially coding and noncoding first exons. In the human genome, ~40% of the genes have completely noncoding first exons and first introns tend to be longer than average (Davuluri et al., 2001).

Stretches of DNA >200 nucleotides with high G+C content and a frequency of CpG dinucleotides close to the expected value are generally known as CpG islands (Gardiner-Garden and Frommer, 1987). As many human

promoters are near CpG islands, Davuluri et al. (2001) classified first exons as CpG related and non-CpG-related based on CpG score. This helped to better characterize the differences in sequence composition between first exons and other regions of the genome. CpG score is defined as the maximum of CpG percentages of all possible sliding windows of length 201 bp within the region of −500 of the transcription start site to +500 of the first donor site. The sequence window that gets the maximum CpG percentage is defined as CpG window. A first exon is CpG related if there exists a CpG window of size 201 with CpG percentage = 6.5. It was estimated that ~70% of the first exons in the human genome are CpG related. FirstEF uses different quadratic discriminant functions to identify CpG related and non-CpG-related first exons, promoters and first donor sites. Refer to *UNIT 4.2* for a brief description on discriminant analysis.

How the algorithm works

FirstEF scans the input DNA for potential first donor sites. During the first step, the program pauses at every GT and computes the a posteriori probability of the donor site, P(donor), by a quadratic discriminant function (donor-QDF), which was trained on donor sites of first exons. If P(donor) = donor cut-off value, FirstEF considers this as a candidate donor site, otherwise it proceeds to next GT.

For every candidate donor site, FirstEF scans a region of length 2000 nt (1500 nt upstream and 500 nt downstream of GT) for the existence of a CpG window of length 201 nt with a CpG score = 6.5. FirstEF decides whether the first exon is CpG-related or non-CpG-related depending on the presence or absence of a CpG window. For more discussion of CpG window and CpG score see Davuluri et al. (2001).

FirstEF uses a sliding window of length 570 nt—considering the first 500 nt as a proximal promoter upstream of transcription start site (TSS) and the following 70 nt as downstream of TSS—within the 1500-nt upstream region of each candidate donor site. FirstEF decides whether the sliding window can be a promoter or not based on the a posteriori probability of promoter, P(promoter). P(promoter) was calculated using two different promoter QDFs (Promoter-QDF), one for CpG-related and the other for non-CpG-related.

If P(promoter) = promoter cut-off value, FirstEF matches the promoter region with the corresponding donor site and evaluates the a posteriori probability of exon, P(exon), by using four different first exon QDFs (exon-QDF). FirstEF reports all those exons with P(exon) = exon cut-off value, along with the promoter region and CpG window, if it exists.

The user can select different cut-off values for donor, promoter, and exon in the range of 0.2 to 1.

Critical Parameters

The critical parameters are the cut-off values of a posteriori probabilities of promoter, exon and donor site. The closer the values of these parameters to 1 the more likely the corresponding predictions are real.

Troubleshooting

The following is a list of some of the more common problems associated with using this software.

1. *Too many predictions*: Increase the cut-off values or consider only those predictions with probability values close to one.

2. *Too few or no predictions*: This may be due to a lack of first exons in the genomic sequence that is being analyzed. However, if the user has strong reason to believe that there exists a first exon or promoter of some gene of interest, experimenting with lower probability cut-off values may produce some predictions.

3. *Promoter of single exon genes*: Single exon genes lack splice site GT and hence FirstEF may fail to predict such promoters. In such cases, lowering cut off value of `P(donor)` would help to predict the promoter region.

4. *First exons with noncanonical splice sites*: FirstEF was trained on first exons with canonical splice site GT. Hence, FirstEF can not predict first exons with noncanonical splice sites. Lowering cut-off value of `P(donor)` might predict a nearby weak donor site and help identifying the promoter region.

Suggestions for Further Analysis

Among the earlier programs, PromoterInspector (Scherf et al., 2000) is the best in locating the gene regulatory regions. CpGpromoter (Ioshikhes and Zhang, 2000) is another promoter prediction program for a large-scale human promoter mapping using CpG islands.

Literature Cited

Burge, C. and Karlin, S. 1997. Prediction of complete gene structures in human genomic DNA. *J. Mol. Biol.* 268:78-94.

Davuluri, R.V., Grosse, I., and Zhang, M.Q. 2001. Computational identification of first exons and promoters in human genome. *Nature Gen.* 29:412-417.

Florea, L., Hartzell, G., Zhang, Z., Rubin, G.M., and Miller, W. 1998. A computer program for aligning a cDNA sequence with a genomic DNA sequence. *Genome Res.* 8:967-974.

Gardiner-Garden, M. and Frommer, M. 1987. CpG islands in vertebrate genomes. *J. Mol. Biol.* 196:261-276.

Ioshikhes, I. and Zhang, M.Q. 2000. Large-scale human promoter mapping using CpG islands. *Nature Gen.* 26:61-63.

Scherf, M., Klingenhoff, A., and Werner T. 2000. Highly specific localization of promoter regions in large genomic sequences by PromoterInspector: A novel context analysis approach. *J. Mol. Biol.* 297:599-606.

Zhang, M.Q. 1997. Identification of protein coding regions in the human genome by quadratic discriminant analysis. *Proc. Natl. Acad. Sci. U.S.A.* 94:565-568.

Key References

Davuluri et al., 2001. See above.

The algorithm details of FirstEF and its accuracy in predicting first exons and promoters using a test set of experimentally known first exons is described. The performance of FirstEF over a large genomic regions (human chromosomes 21 and 22) is also discussed.

Davuluri, R.V., Grosse, I., and Zhang, M.Q. (Submitted)

FirstEF has been used to perform an initial computational annotation of the promoters and the first exons for all 24 human chromosomes. Visit http://genemap.med.ohio-state.edu for accessing the annotations.

Internet Resources

http://rulai.cshl.org/tools/FirstEF

The FirstEF Web-based version.

http://genemap.med.ohio-state.edu

The Bioinformatics Unit of the Human Cancer Genetics Program at The Ohio State University. The First Exon genome browser is available from this site.

http://www.ncbi.nih.gov/BLAST

The BLAST and MEGABLAST homepage at the NCBI. See UNITS 3.3 & 3.4 for more information.

http://pbil.univ-lyon1.fr/sim4.html

The SIM4 Web site. See Florea et al. (1998) for more information.

http://genes.mit.edu/GENSCAN.html

The GENSCAN server at MIT.

http://www.cshl.edu/mzhanglab or http://rulai.cshl.edu

The Zhang Laboratory Computational Biology and Bioinformatics website. A link to MZEF is available through this site.

http://www.wormbase.org/db/seq/frend

The Sequence Feature Renderer home page. Figure 4.7.3 was created using this tool.

Contributed by Ramana V. Davuluri
Ohio State University
Columbus, Ohio

Using N-SCAN or TWINSCAN to Predict Gene Structures in Genomic DNA Sequences

Marijke J. van Baren,[1] Brian C. Koebbe,[1] and Michael R. Brent[1]

[1]Washington University, St. Louis, Missouri

ABSTRACT

N-SCAN is a gene-prediction system that combines the methods of ab initio predictors like GENSCAN with information derived from genome comparison. It is the latest in the TWINSCAN series of programs. This unit describes the use of N-SCAN to identify gene structures in eukaryotic genomic sequences. Protocols for using N-SCAN through its Web interface and from the command line in a Linux environment are provided. Detailed discussion about the appropriate parameter settings, input-sequence processing, and choice of genome for comparison are included. *Curr. Protoc. Bioinform.* 20:4.8.1-4.8.16. © 2007 by John Wiley & Sons, Inc.

Keywords: N-SCAN • TWINSCAN • gene prediction • sequence alignment • comparative genome analysis • cross-species sequence comparison • genome annotation

INTRODUCTION

N-SCAN is a gene-structure prediction system for eukaryotic genomic sequences. N-SCAN is the latest in the TWINSCAN series of programs. It takes as input a sequence to be annotated (the target sequence) and a multiple sequence alignment of the target sequence and one or more closely related genomes (the informant genomes). N-SCAN models the pattern of tolerated substitutions, insertions, or deletions in the aligned sequences and uses a probability model to combine this information with information from patterns in the target DNA sequence. The model exploits the fact that features such as introns, UTRs, coding sequence, and splice sites all exhibit characteristic patterns in the target sequence and all evolve under distinct selective pressures, leaving an imprint on local patterns of conservation. Since N-SCAN uses a combined model, it can predict genes that are strongly indicated by the target DNA sequence even if they are not well conserved in the informant genome(s).

N-SCAN can be run either through a Web browser pointed at the N-SCAN server (*http://mblab.wustl.edu/nscan/submit/*) or through the command line on a local computer. The Web server is recommended for all users except those with substantial bioinformatics experience and the need to (1) process >1 Mb per day on a sustained basis, (2) use N-SCAN outside the supported clades, or (3) use N-SCAN on proprietary sequences that cannot be stored on the server. To use the Web server, the user is required to register. A personal account is created so that the user can track the progress and results of all gene predicion submissions.

The Basic Protocol describes using the Web interface of N-SCAN to find gene structures, exons, coding sequences, and protein sequences. The Support Protocol describes how to obtain and install N-SCAN software on a local computer running Linux; Alternate Protocol 1 describes how to run N-SCAN on such a computer. Alternate Protocol 2 describes the use of a Perl script, which can automate the operation of N-SCAN on a local computer.

Current Protocols in Bioinformatics 4.8.1-4.8.16, December 2007
Published online December 2007 in Wiley Interscience (www.interscience.wiley.com).
DOI: 10.1002/0471250953.bi0408s20
Copyright © 2007 John Wiley & Sons, Inc.

N-SCAN requires genomic sequences from at least two organisms of appropriate evolutionary distance. Currently, N-SCAN is available for sequences from the following clades: *Mammalia* (mammals), *Caenorhabditis* (a genus of roundworms including *C. elegans* and *C. briggsae*), *Drosophila* (fruit files) and *Cryptococcus neoformans* (a species of pathogenic fungus including strains JEC21 and H99). An earlier version of N-SCAN, TWINSCAN, is available for *Brassicaceae* (mustard family, including *Arabidopsis thaliana*), *Zea* (maize), and *Oryza* (rice). For all these clades, the N-SCAN Web server provides the appropriate informant database.

N-SCAN is tuned to predict the 5′-untranslated region and the coding portions of genes as accurately as possible, with as few false positives as possible. It is designed to annotate large sequences (such as whole genomes) automatically. N-SCAN cannot predict multiple splices for a single gene and is not tuned for accurate prediction of polyadenylation or promoter signals.

BASIC PROTOCOL

USING THE N-SCAN WEB SERVER

When N-SCAN is run through the Web server, the user selects the clade and species from which the target sequence is derived. Once a selection is made, an informant database is automatically selected and displayed on the Web page. The user provides the target sequence, either by cutting and pasting from another window or by uploading a sequence file. When the user submits the job by clicking the Predict Genes button, a submission ID is generated and the user is taken to a page that lists the status of the submission. When processing is complete, this page changes to display the results, and an email notification is sent to the user. This email contains a link to the results page, which will be available until the user deletes the project.

Necessary Resources

Hardware

Computer with an Internet connection

Software

Internet Explorer version 6.0 or higher, Firefox v. 1.5 or higher, or Safari v. 1.3 or higher

Files

The N-SCAN server requires genomic DNA sequence as input. Each input should contain a single sequence consisting of the letters ACGTN in upper, lower, or mixed case. All other characters, including whitespace characters such as line breaks, tabs, and carriage returns are ignored. Optionally, the input may begin with a FASTA (*APPENDIX 1B*) file header, i.e., a single line beginning with > followed by a sequence name, comments, and a carriage return. The header has no size limit.

Submit target sequence to N-SCAN

1. Register on the N-SCAN page at *http://mblab.wustl.edu/nscan/register*.

   Fill in your preferred username, your email address, your first and last name, and your institutional affiliation. Press the Create account button at the bottom of the green box. An account is created for you, and all your predictions will be kept on the N-SCAN server until you delete them. A new page appears, telling you that your password is being emailed to you. Check your email to retrieve the password.

   By default, gene prediction results are accessible on the N-SCAN output page, from which they can be downloaded. If you want to have the results emailed to you, after logging in, go to Preferences and check the appropriate box.

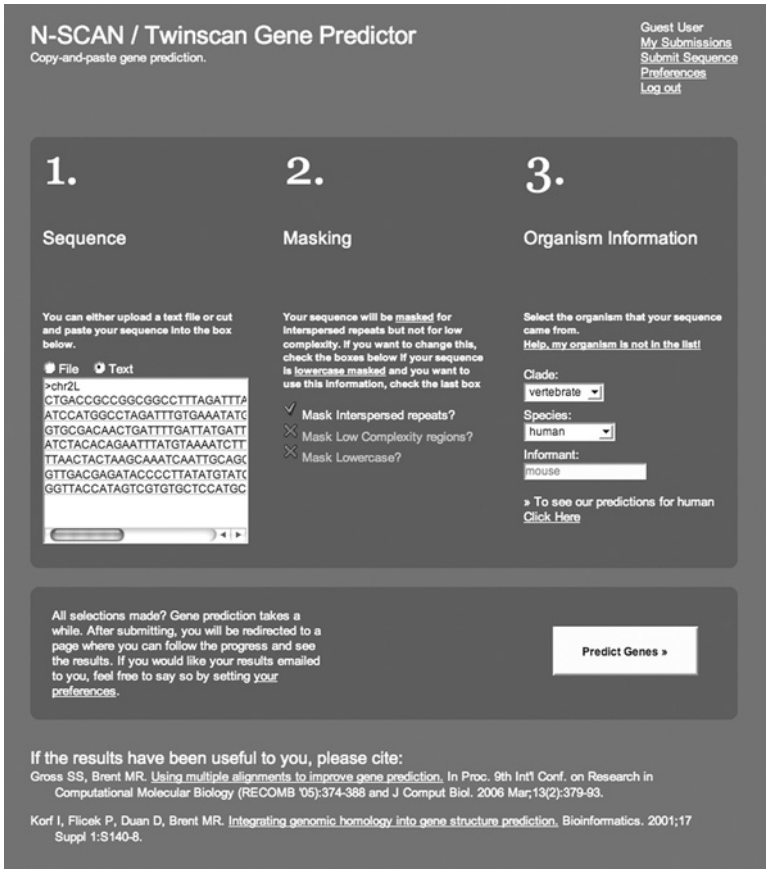

N-SCAN / Twinscan Gene Predictor
Copy-and-paste gene prediction.
Guest User
My Submissions
Submit Sequence
Preferences
Log out
1.
Sequence
2.
Masking
3.
Organism Information
You can either upload a text file or cut and paste your sequence into the box below.
File
Text
>chr2L
CTGACCGCCGGCGGCCTTTAGATTTA
ATCCATGGCCTAGATTTGTGAAATATC
GTGCGACAACTGATTTTGATTATGATT
ATCTACACAGAATTTATGTAAAATCTT
TTAACTACTAAGCAAATCAATTGCAGC
GTTGACGAGATACCCCTTATATGTATC
GGTTACCATAGTCGTGTGCTCCATGC
Your sequence will be masked for interspersed repeats but not for low complexity. If you want to change this, check the boxes below if your sequence is lowercase masked and you want to use this information, check the last box
Mask Interspersed repeats?
Mask Low Complexity regions?
Mask Lowercase?
Select the organism that your sequence came from.
Help, my organism is not in the list!
Clade:
vertebrate
Species:
human
Informant:
mouse
» To see our predictions for human
Click Here
All selections made? Gene prediction takes a while. After submitting, you will be redirected to a page where you can follow the progress and see the results. If you would like your results emailed to you, feel free to say so by setting your preferences.
Predict Genes »
If the results have been useful to you, please cite:
Gross SS, Brent MR. Using multiple alignments to improve gene prediction. In Proc. 9th Int'l Conf. on Research in Computational Molecular Biology (RECOMB '05):374-388 and J Comput Biol. 2006 Mar;13(2):379-93.
Korf I, Flicek P, Duan D, Brent MR. Integrating genomic homology into gene structure prediction. Bioinformatics. 2001;17 Suppl 1:S140-8.

Figure 4.8.1 The N-SCAN Web server at *http://mblab.wustl.edu/nscan*. If you are not registered, a Register link will be visible in the top right corner.

It is possible to do an example N-SCAN run before registering. Go to http://mblab.wustl.edu/nscan/submit. You will see a page very similar to Fig 4.8.1, but the Predict Genes button is replaced with See an example of Twinscan/N-SCAN in action! Clicking on this link starts an example run, which selects a Drosophila melanogaster sequence and runs N-SCAN with default settings. The sequence contains 6 genes, which are all predicted correctly.

2. Log onto the server at *http://mblab.wustl.edu/nscan/login*.

   Type in your username and password, and press the Login button. The N-SCAN Web page appears (Fig. 4.8.1). If your browser allows cookies to be set, you will be logged in automatically the next time you go to the N-SCAN page.

3. Input the target sequence. There are two possible ways to submit a sequence to N-SCAN:

   a. Select Text, copy the target sequence, and paste it into the input box (see Fig. 4.8.1).

   b. Load the target sequence from a file by selecting File and clicking the Browse button that appears next. A dialog box will pop up. Users can select any file on their computer from this dialog box.

   In either case, the input must be a nucleotide sequence file consisting of upper or lower case ACGTN. If the first line begins with >, it will be treated as a sequence identifier and may contain any characters. All other lines may only contain ACGTN in upper, lower, or mixed case, whitespace, and numbers. This is useful for pasting in files that contain

line numbers and other formatting information. However, if N-SCAN detects any other character, a warning is issued, informing the user that the sequence is not valid.

The N-SCAN Web server only accepts sequences between 500 bp and 2 Mb in length. To annotate entire large genomes, it is most efficient to download and run N-SCAN locally or to contact the authors about collaboration.

4. Select the type of sequence masking.

   Before gene prediction, the sequence should be masked for interspersed repeats. These repeats are degenerate copies of transposable elements, and make up about a third of the human genome. The coding portion of genes almost never contains interspersed repeats, therefore, masking results in better gene prediction. By default, a sequence is masked for interspersed repeat.

   Low complexity and simple repeats are short, repetitive sequences such as TATATATATA or GAGATAGAGAGA. Genes sometimes contain such repeats, so by default they are not masked. Check the Mask Low Complexity regions checkbox to change this.

   The user can mask out additional sequence by inputting the sequence to be masked in lowercase and checking the Mask Lowercase checkbox. All boxes can be checked independently.

5. Select the clade and organism of the target.

   Under Clade is a drop-down menu with a list of clades: nematode fungus, vertebrate, insect, and plant. Left-click the mouse button on the Clade box to display the list of options, then move the cursor to the appropriate option and left-click the mouse button again. Now select the species in the Species box in the same manner. This list is constantly updated with newly sequenced genomes. When the organism is selected, N-SCAN automatically selects the informant genome and displays it in the Informant box. The current version of N-SCAN automatically chooses mouse as the informant organism for human target sequences, human as informant for a nonhuman mammal target, Brassica oleracea for an Arabidopsis target, Arabidopsis thaliana for a non-Arabidopsis dicot plant target, C. briggsae for a C. elegans target, C. elegans for a target that is a member of the genus Caenorhabditis other than C. elegans, Drosophila ananassae for Drosophila melanogaster, and Drosophila melanogaster for all other insects.

   If N-SCAN annotations for the target species are available on the UCSC genome browser (UNIT 1.4), a link to these annotations will appear as well. If the input sequence is taken from a genome assembly, following this link is recommended.

   If the target species is not listed, select the organism that is closest. Try several targets in different runs, to see which one works best.

6. Run N-SCAN.

   After specifying the input sequences, the masking options, and the target organism, click the Predict Genes button in the lower dark green box (under the sequence box) to begin processing. The browser now displays a Submission page that contains information on your input (Fig. 4.8.2). The submission has an I.D. number, listed at the top of the page. Below the target species, the current state of your job is shown, which can be Queued, Masking, Aligning, Predicting, or Complete. The current status of the job is highlighted, and an explanation of what it means is shown under Status. When N-SCAN is finished, this page displays your results and a link is emailed to you.

   At the top right side of the Submissions page is a link called My Submissions. When you click on My Submissions, you will be taken to a page displaying all your previous jobs. If you click on a number, the results page for that job is shown.

Examine results on the N-SCAN output Web page

7. Check the run parameters on the Submissions page.

   The N-SCAN output page is shown in Figures 4.8.3 and 4.8.4. Click the Submission Details button to see the target and informant sequences, the N-SCAN version, and links to the input sequence and parameter files. This information can be useful for ensuring

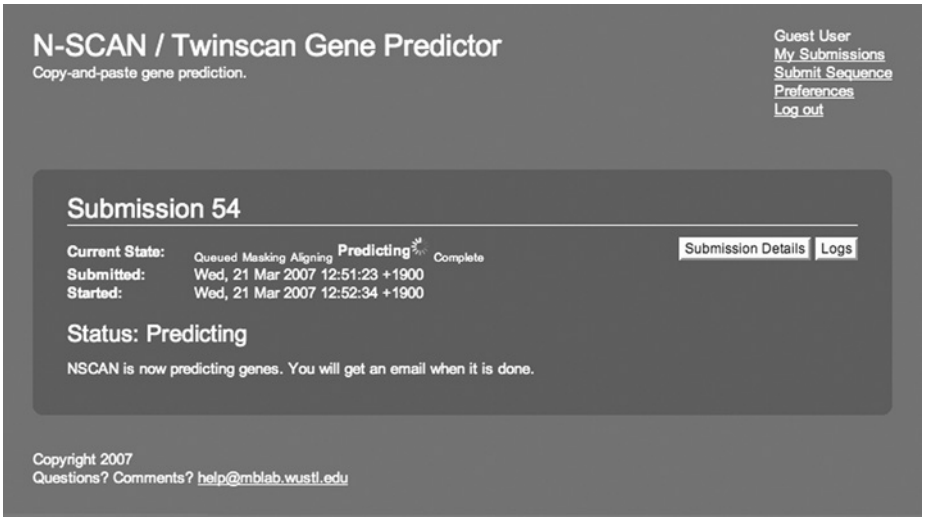


Figure 4.8.2 The Submission page that appears when a sequence has been submitted to the Web server. The current status of the job is explained at the bottom.

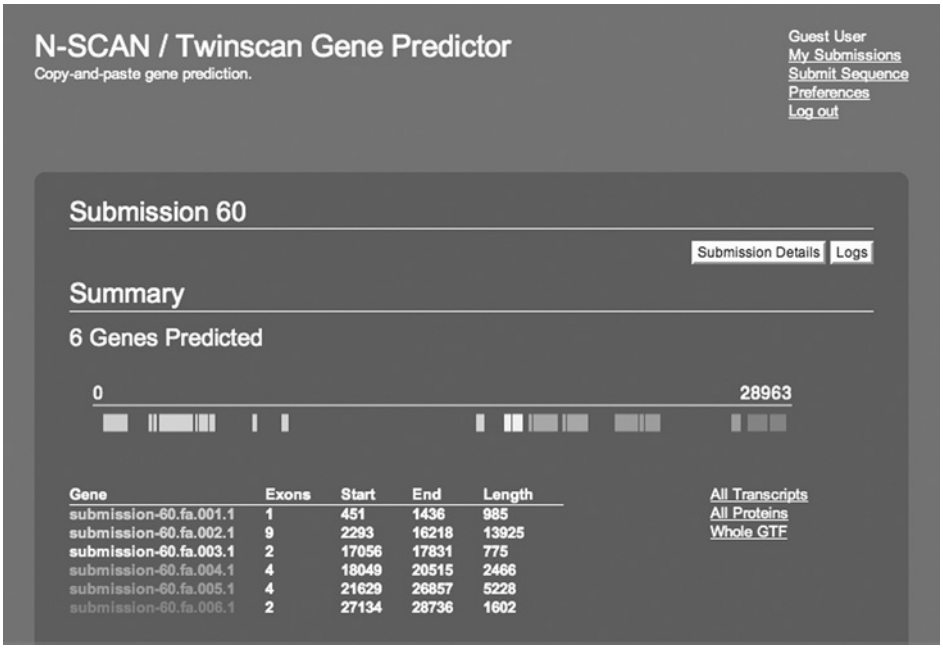


Figure 4.8.3 An example of the top portion of the Submission Results Web page that automatically replaces the waiting page when N-SCAN has completed processing of a sequence submitted to the server.

that N-SCAN was run as expected, and for reproducing results. Changes to any of these specifics can change the results.

8. Look at the schematic summary of predicted genes.

   The top of the page (Fig. 4.8.3) displays the number of genes predicted by N-SCAN and a graphical representation of the predicted genes on the input sequence. A thin line represents the input sequence (with the length shown on the right side), and exons are shown as colored boxes below this line. Exons with the same color are part of the same gene. A short summary of each gene is listed below the picture.

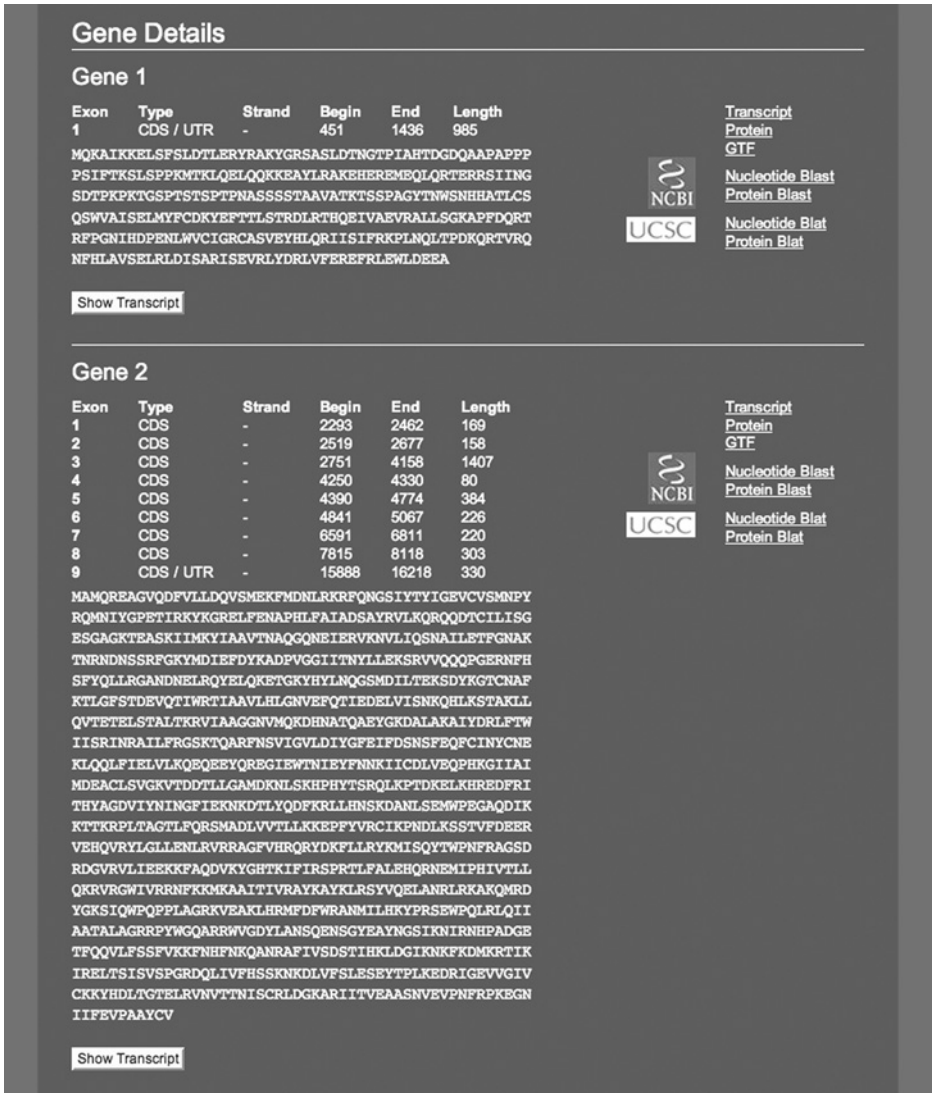


Figure 4.8.4 An example of the bottom portion of the Submission Results Web page.

9. Look at textual details and sequences for predicted genes.

   Under the gene overview table, details on every gene are listed (Fig. 4.8.4). A table shows the type, orientation (positive strand or negative strand), start position, end position, and length of each predicted exon. Noncoding exons are called UTR (for untranslated region), protein coding exons are CDS. The first coding exon usually contains an untranslated region, so this exon has both types; it is listed as UTR/CDS. In a later version, 3′ UTRs will be included in the output.

   Below the exon table, you will find the predicted protein. Click on the link to the NCBI BLAST page to submit this protein or the corresponding transcript to NCBI's BLAST server (UNITS 3.3 & 3.4). This is useful for checking whether the predicted gene is a known gene or is homologous to known genes. If the UCSC genome browser (UNIT 1.4) lists the target species of this submission, a second link will be shown. Click on this link to align your sequence to its genome in the genome browser. This allows you to see neighboring genes, as well as many other gene predictions and annotation tracks in the region.

   To see the predicted transcript, click on the Transcript button below the protein sequence.

10. Download output.

Gene predictions can be downloaded as transcript sequences, protein sequences, or GTF files. GTF stands for gene transfer format, one of the standard formats for genome annotation. GTF is derived from the GFF format developed at the Sanger Center. This file can be examined manually to obtain exact coordinates for one or two genes of particular interest, but it is intended primarily for automated processing.

Each line of the GTF has the following fields separated by tabs:

[seqname] [source] [feature] [start] [end] [score] [strand] [frame] [attributes]

The [feature] field consists of one the words `start_codon`, `stop_codon`, `5UTR`, *or* `CDS`. *The attributes field contains the word* `gene_id` *followed by an automatically generated id symbol, then a semicolon, the word* `transcript_id`, *and another automatically generated symbol. Coding regions and other features with the same transcript id belong to the same transcript. The inclusion of both* `gene_id` *and* `transcript_id` *allows for alternative splices of a single gene, but that feature is not currently used by N-SCAN.*

More information on the GTF format can be found at http://mblab.wustl.edu/GTF22.html.

To download all predictions, click one of the links at the top of the page, next to the gene overview table. To download the sequence or coordinates for any one gene, click on the appropriate link next to its exon table.

11. Manage your submissions.

Click on My Submissions at the top right corner of the Submission or N-SCAN input page. This will show an overview of all your submissions (Fig. 4.8.5). From here, you can access the Submission page of each of your jobs by clicking on its Submission link. The overview also lists the status of your running jobs, and, when they are complete, a link to the GTF. Press the Delete button if you no longer need the results.

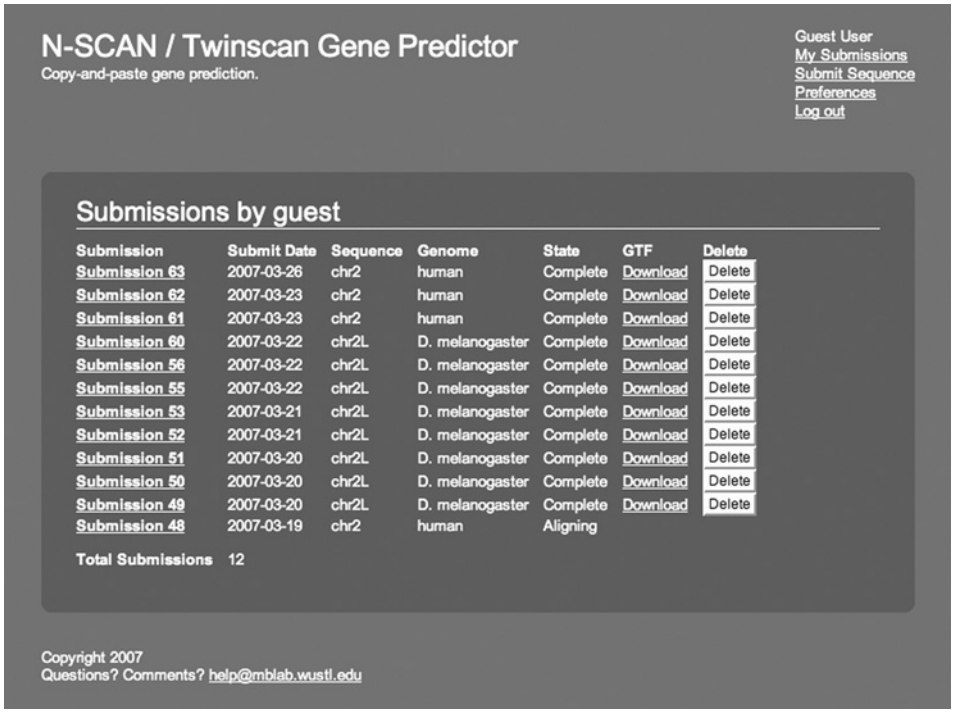

N-SCAN / Twinscan Gene Predictor
Copy-and-paste gene prediction.
Guest User
My Submissions
Submit Sequence
Preferences
Log out
Submissions by guest
Submission | Submit Date | Sequence | Genome | State | GTF | Delete
Submission 63 | 2007-03-26 | chr2 | human | Complete | Download | Delete
Submission 62 | 2007-03-23 | chr2 | human | Complete | Download | Delete
Submission 61 | 2007-03-23 | chr2 | human | Complete | Download | Delete
Submission 60 | 2007-03-22 | chr2L | D. melanogaster | Complete | Download | Delete
Submission 56 | 2007-03-22 | chr2L | D. melanogaster | Complete | Download | Delete
Submission 55 | 2007-03-22 | chr2L | D. melanogaster | Complete | Download | Delete
Submission 53 | 2007-03-21 | chr2L | D. melanogaster | Complete | Download | Delete
Submission 52 | 2007-03-21 | chr2L | D. melanogaster | Complete | Download | Delete
Submission 51 | 2007-03-20 | chr2L | D. melanogaster | Complete | Download | Delete
Submission 50 | 2007-03-20 | chr2L | D. melanogaster | Complete | Download | Delete
Submission 49 | 2007-03-20 | chr2L | D. melanogaster | Complete | Download | Delete
Submission 48 | 2007-03-19 | chr2 | human | Aligning
Total Submissions 12
Copyright 2007
Questions? Comments? help@mblab.wustl.edu

Figure 4.8.5 An example of a My Submissions page. From here, all previous and running jobs can be accessed using the links on the left.

RUN N-SCAN FROM THE COMMAND LINE ON A LOCAL COMPUTER

Running N-SCAN on a local computer, as opposed to running it via the Web server (see Basic Protocol), is recommended for users with bioinformatics experience who need to (1) process >1 Mb per day on a sustained basis, (2) use N-SCAN outside the supported clades, or (3) use N-SCAN on proprietary sequences.

The N-SCAN software package includes the N-SCAN executable itself, together with supporting Perl scripts. The inputs to the N-SCAN executable include a target sequence to be annotated, a parameter file, and an alignment between the target and informant sequence(s). Two supporting scripts, `lav2maf` and `maf_to_align.pl`, create the alignment file from a Blastz report.

Another script, `Nscan_driver.pl`, implements a simple four-step "pipeline" for automatically masking the target sequence, running `Blastz`, running `lav2maf` and `maf_to_align.pl`, and running N-SCAN. Since `Nscan_driver.pl` (see Alternate Protocol 2) may require modification for a given user's needs and environment, this protocol focuses on how to execute these four steps manually. Once the manual procedure is understood, users will be in a better position to understand whether `Nscan_driver.pl` will work unmodified in their environment and if not, what modifications are required. Alternate Protocol 2 describes the use of `Nscan_driver.pl`, but the authors strongly recommend reading Alternate Protocol 1 first.

ALTERNATE PROTOCOL 1

Preparing Data Files and Running N-SCAN Manually

Starting with the data files described below, there are five steps to preparing intermediate files and running N-SCAN. These five steps can also be orchestrated through a Perl script called `Nscan_driver.pl` (see Alternate Protocol 2).

NOTE: Users who are unfamiliar with Unix are encouraged to read *APPENDIX 1C*, which provides guidance for working in a Unix environment.

Necessary Resources

Hardware

A computer on which N-SCAN has been installed and tested (see Support Protocol), with at least with a 2-GB memory and a processor whose computing speed is at least equivalent to a 2 GHz ×86 processor. Free disk space should be at least five times the combined size of the uncompressed target and informant sequences. For example, 2 GB is recommended for *Arabidopsis*, using the current *Brassica* database as informant, while 30 GB is recommended for a pair of assembled mammalian genomes.

Software

An N-SCAN software distribution. See Support Protocol for obtaining and installing N-SCAN.

Perl v. 5.8.5 or later (*http://www.perl.com*). Perl is already available on most Linux systems. To check whether Perl is available and if so, which version, enter the following command at the Unix shell prompt: `perl -v`. If Perl is not available, those without substantial Unix experience, should consult their system administrator.

RepeatMasker (v. 10/6/2006 or later is recommend, also see *UNIT 4.10*; obtain RepeatMasker from *http://www.repeatmasker.org/RMDownload.html*) N-SCAN can be used without repeat masking, but speed, accuracy, and disk usage will be affected.

Blastz (v. 12/27/2004 or later), and multiz (4/28/2005 or later), which can be downloaded from *http://www.bx.psu.edu/miller_lab/*. The multiz package

includes the `lav2maf` program, which you will need for converting the blastz output.

Files

The target sequence to be annotated, and the informant sequence(s) in FASTA format (*APPENDIX 1B*). The informant sequences can be shotgun reads or assemblies.

A parameter file. For organisms available on the website, parameter files are included in the parameters directory of the N-SCAN distribution. See Commentary for a discussion of parameter estimation and substitution.

1. Mask repeats in the target sequence by replacing them with N's.

   N-SCAN's accuracy is somewhat improved by replacing sequences that are unlikely to contain genes of interest with N's. Such sequences include mobile repetitive elements and functional RNA genes such as tRNAs. The effect of repeat masking is greater the more repetitive the genome, but even in relatively nonrepetitive genomes like that of C. elegans, masking repeats yields small improvements. If RepeatMasker is installed, repeat masking is achieved by the command:

   `RepeatMasker` [-options] [target-sequence file]

   This command creates several files, including a masked sequence. This sequence has the same name as the input sequence, followed by `.masked`. *Use this file in all the commands described below.*

   To view the current masking options, type RepeatMasker with no arguments. If the species or clade of interest is not listed, it is possible to specify the name of a repeat library file directly. For more information on how to run RepeatMasker, see UNIT 4.10 or run

   `RepeatMasker –h`

   There is also a RepeatMasker Web server at http://www.repeatmasker.org/cgi-bin/WEBRepeatMasker.

2. Create an alignment file using the informant sequences.

   The informant-alignment file consists of a FASTA header line and a line for each informant. The length of each informant line is the same as the length of the target sequence. For each character in the target sequence, there is a corresponding character in each informant sequence. The informant sequence alphabet is {`A,C,G,T,_,.`} *where an informant character from the set* $\{A, C, G, T\}$ *means the informant character aligns to the corresponding target characters,* {`_`} *is used for informant gaps within aligned regions, and* {`.`} *is used for target regions to which the given informant does not align. To create the alignment sequence for a single informant, run Blastz and convert its output using the following commands (the greater-than sign before the output file name is interpreted by Unix as specifying where the output should be stored):*

   `blastz` [masked target sequence] [informant sequence] > `target.lav`

   `lav2maf target.lav` [masked target sequence] [informant sequence] > `target.maf`

   `maf_to_align.pl` [output directory] `target.maf ascending` [masked target sequence] [informant sequence] > `target.align`

   The program `maf_to_align.pl` *is included in the N-SCAN download package.*

   For multiple informants, maf files can be generated using multiz. For running multiz, follow instructions from http://www.bx.psu.edu/miller_lab/. In addition, multiz alignments in maf form, generated by the bioinformatics group at UCSC, are available for download at http://hgdownload.cse.ucsc.edu/downloads.html.

3. Run the N-SCAN executable by entering the following command on a single line:

   `nscan −o` [parameter file] [masked target sequence] `−a=target.align >` [output file]

   The output file is in GTF format.

   It is possible to run the N-SCAN code in TWINSCAN mode, by omitting the –a option, using the –c option with a conservation sequence file, and using a TWINSCAN-specific parameter file. For generating a conservation sequence and more details on TWINSCAN, see the README provided with the N-SCAN package.

ALTERNATE PROTOCOL 2

Using `Nscan_driver.pl` on a Local Computer

The three steps described in Alternate Protocol 1 can be run automatically using a Perl script called `Nscan_driver.pl`, which is included in the `/bin` directory of the N-SCAN distribution. However, this script may not satisfy the needs of all users, as it does not give access to all options available in the N-SCAN code. The script needs a configuration file that lists the full paths to all programs run by `Nscan_driver.pl`. Users will need to customize this file for their particular applications. It may be necessary to adapt `Nscan_driver.pl` to the specific environment of the user's system, but this advanced procedure is beyond the scope of this unit.

For necessary resources, see Alternate Protocol 1.

To run the Perl script, change to the directory `N-SCAN/bin`. Next, run the N-SCAN pipeline by entering the following command on a single line:

`./Nscan_driver.pl` [masked target sequence] [configuration file]

where [masked target sequence] is the sequence to be annotated (see Alternate Protocol 1 for instructions on file format), and [configuration file] contains the full paths to all programs. An example configuration file can be created by running

```
Nscan_driver.pl -config > configfile
```

The resulting config file must be edited to reflect the local configuration.

The output files are created in the current directory by default. To specify an alternate directory, include `-d = [output directory]`. The `Nscan_driver.pl` script will generate files whose names consist of the target sequence file name with the following extensions: `.masked`, `.align`, and `.gtf`. If any of these files already exists, the script will print a warning and stop running. Delete or move the existing files before rerunning.

The `Nscan_driver.pl` program will mask the target sequence, create a `.align` format file using `Blastz` and the file format conversion scripts, and run N-SCAN. If the sequence is already masked, the command can be run as follows:

`./Nscan_driver.pl --nomask` [target sequence] [configuration file]

If a `.align` file already exists for the target sequence, run

`./Nscan_driver.pl --blastz =` [align file] [target sequence] [configuration file]

For additional options, run `Nscan_driver.pl` with no arguments.

OBTAINING AND INSTALLING N-SCAN ON A LOCAL COMPUTER

N-SCAN 4.0 is open source software. Local copies of N-SCAN (including source code if needed) can be obtained under a free license from Washington University, through *http://mblab.wustl.edu/software/*. This protocol describes how to obtain the N-SCAN distribution and install it on the most commonly used type of Linux computer. N-SCAN can also be compiled from source code for machines based on other architectures or running other versions of the Unix operating system, but these advanced procedures are beyond the scope of this unit.

Necessary Resources

Hardware

Computer with CPU based on Intel ×86 architecture and running the Linux operating system (at least 2-Gb memory, 1-GHz processor, and 100 Mb of free disk space are recommended)

Software

N-SCAN software distribution. The distribution can be downloaded from *http://mblab.wustl.edu/software/* by clicking the N-SCAN – Latest Version link.

1. Obtain the N-SCAN distribution file and place it in the directory where N-SCAN should be installed.

   The user must have read, write, and execute permission in this directory.

2. While in the directory containing the distribution, enter the following command at the Unix prompt:

   `tar −xvzf`[distribution-file-name]

   If this command is successful, it creates a directory called N-SCAN.

3. Test the installation by changing to the N-SCAN directory and entering the following command at the prompt:

   `./test-executable`

   If this test is successful, N-SCAN can be run as described in Alternate Protocol 1.

   Otherwise, a message will appear indicating that the machine is not compatible with the N-SCAN executable and offering further suggestions.

GUIDELINES FOR UNDERSTANDING RESULTS

Analyzing Problems and Errors

If the Web server finds a problem with the input sequence, it will generally provide an immediate explanation of the problem. Mammalian sequences may take 2 to 4 hr to run, during which there is no response. Heavy traffic on the server may cause any job to wait many hours before running. If there is no response 24 hr after submission, write to *nscan@mblab.wustl.edu* and include the submission I.D. for the job in question.

If unexpected results are received, such as gene density that is much higher or lower than expected, the first step is to check that the correct sequence was submitted. One useful check is to compare the length of the sequence N-SCAN processed, which is returned at the top of the Submission results page, with the length of the sequence you intended to submit. If the sequence is correct, verify that the correct target organism was selected, and that the masking settings were as intended. This information can be found under the Submission Details button on the Submission results page. In addition, check the program

logs under the Logs button to see if any of the procedures returned an error. Questions and comments regarding unexpected results may be sent to *nscan@mblab.wustl.edu* in a message that includes the job I.D. Do not forget that unexpected results are sometimes correct.

COMMENTARY

Background Information

Approaches to gene structure prediction

Currently, there are two major approaches to automated gene prediction. De novo systems take one or more genomes as input while expression-based systems use databases of known transcripts to annotate a genome.

Single genome ab initio (or de novo) gene predictors take only the target genome as input, and a probabilistic model that abstracts patterns common to many genes is used to annotate the genome. Examples of single de novo gene predictors are GENSCAN (Burge, 1997; Burge and Karlin, 1997), FGeneSH (Salamov and Solovyev, 2000), and Augustus (Stanke and Waack 2003; Stanke et al., 2006). A second approach to de novo gene prediction, typified by TWINSCAN/N-SCAN (Korf et al., 2001; Flicek et al., 2003; Gross and Brent, 2006), SLAM (Alexandersson et al., 2003), and SGP2 (Parra et al., 2003), augments a de novo system with information based on alignments between the target genome and one or more related informant genome(s). Such a multi-genome de novo system is generally better than a similar system run with only one genome. For any given genome, however, a well-optimized single-genome predictor can equal or surpass a poorly optimized dual-genome predictor.

Annotation pipeline methods for gene prediction use databases of known proteins and/or cDNA sequences and map these to the target genome, in an attempt to predict genes whose protein products would be similar to those of known genes. Subsequently, de novo annotation is attempted for regions that have no matching sequences. The best known expression-based system is Ensembl (Hubbard et al., 2007). Such systems have the advantage of providing a single, discrete piece of evidence for each prediction (namely, similarity to a specific cDNA or its translation), and hence are sometimes called "evidence-based" (Mouse Genome Sequencing Consortium et al., 2002). However, the mapping problem is difficult, so accuracy is not perfect even for close relatives of known genes. Furthermore, these pipelines still need a reliable de novo predictor for unmatched regions, or leave them unannotated.

Accuracy measures

The accuracy of gene predictors can be assessed by their ability to predict individual coding nucleotides, exons (either exactly or approximately), and complete genes, and also by a variety of other measures (Guigo et al., 2000). For all these measures, it is useful to consider sensitivity and specificity separately. Sensitivity is the number of correctly predicted features as a fraction of the true number in the sequence. Specificity is the number of correctly predicted features as a fraction of the number predicted. For example, exact gene sensitivity is the number of genes predicted exactly right (start codon, stop codon, and all splice sites in between) divided by the number of genes present in the sequence. Different measures of accuracy are appropriate for different applications. For example, in a project aimed at cloning thousands of full-length open reading frames, pursuing genes whose boundaries have been miscalled is expensive. Thus, a gene predictor with high specificity for both the start and stop codons is appropriate. At the nucleotide and exon levels, there is often a sensitivity-specificity tradeoff, with the most sensitive predictor doing poorly on specificity, and vice versa. For example, sensitivity at the nucleotide level can be increased (at the cost of specificity) by simply classifying randomly selected nucleotides as coding. Some of these will in fact be coding nucleotides, increasing sensitivity, while others will not, decreasing specificity. Predicting a complete gene correctly by such a simple procedure, however, is extremely unlikely. For example, either overpredicting or underpredicting of starts or stops will limit both sensitivity and specificity on the exact-gene level. As a result, specificity and sensitivity on the gene level tend to be positively correlated.

In general, gene predictors are most accurate at the nucleotide level, less accurate at the exact exon level, and least accurate at the exact gene level. At the exon level, initial and terminal exons are predicted with less accuracy than internal exons: in humans, N-SCAN

predicts 59% of initial exons and 66% of terminal exons exactly right, versus 89% of internal exons. This is one of the reasons that gene predictors can identify most coding nucleotides and even most coding exons, but finding correct gene structures is still a challenge. This is especially true in larger genomes, in which less of the sequence is coding. As a result, gene predictors tend to differ from one another least in their nucleotide accuracy, more in their exon accuracy, and most of all in their exact gene accuracy.

Accuracy on specific genomes

For many genomes, N-SCAN is currently the best available de novo gene predictor (see below). However, its accuracy is dependent on optimization for every genome for which it is used. The Brent laboratory recently created a free software tool that automates this modeling procedure, called iParameterEstimation (*http://mblab.wustl.edu/software/iparameterestimation*). In this section, brief comments and impressions are presented with regard to each of the genomes for which N-SCAN has been optimized. In all cases, the authors of this unit are actively working on enhancing the underlying genome models, so significant accuracy improvements are expected in the future.

Mammals. At the time this unit was written, N-SCAN appears to be the most accurate de novo gene predictor for mammalian sequences (Guigo et al., 2006; Stanke et al., 2006). In particular, it is much more specific than other systems with comparable sensitivity (Gross and Brent, 2006; Guigo et al., 2006). Further, N-SCAN is particularly good at the extremely challenging problem of exact-gene prediction, relative to other systems to which it has been compared. Nucleotide sensitivity is in the 90% to 95% range, exact-exon sensitivity in the 75% to 85% range, and exact-gene sensitivity in the 30% to 40% range. Specificities are probably in the same ranges. It is easier to measure sensitivity than specificity because sensitivity can be calculated on a partially annotated genome. In contrast, all genes should be known to measure specificity correctly, and currently no genome is annotated entirely. Therefore, the specificity measure is always an underestimation.

C. elegans. Current gene finders are more accurate on *Caenorhabditis* genomes than on mammals by roughly 5% to 10% on the nucleotide and exon levels and 25% to 30% on the exact-gene level. TWINSCAN, the predecessor of N-SCAN, appears to be more accurate than GeneFinder by ∼13% on the gene level, due mainly to better detection of gene boundaries and greater specificity at the exon and nucleotide levels (Wei and Brent, 2006). N-SCAN's results are comparable to TWINSCAN's (R.H. Brown and M.R. Brent, unpub. observ.).

Drosophila species. N-SCAN is capable of using multiple informant genomes for a single target. In the case of *Drosophila melanogaster*, using three other insect species (*D. yakuba*, *D. pseudoobscura*, and *A. gambia*) improved the accuracy of gene prediction, to 55% gene sensitivity, 77% exon sensitivity, and 93% nucleotide sensitivity (R.H. Gross and M.R. Brent, unpub. observ.). A later run included *D. ananassae* rather than *A. gambia* and improved exact gene sensitivity by ∼5% to 60% (R.H. Brown and M.R. Brent, unpub. observ.).

Arabidopsis thaliana. The state of the art in *Arabidopsis* gene finding is even better than for worms, although the difference is not large. TWINSCAN appears to be much more accurate than GlimmerM, GeneMark.hmm, GENSCAN, and GeneSplicer (Allen et al., 2004, and references therein). N-SCAN has not been optimized for *Arabidopsis*.

Factors affecting accuracy

Although accuracy on any genome or clade can be improved by tailored modeling and fine tuning, gene finding seems to be inherently harder in some genomes than others. This is not well understood, but intron length and average number of introns per transcript are significant factors. In general, prediction is more accurate in compact genomes like those of *C. elegans* and *A. thaliana* than in big genomes with long introns and large numbers of pseudogenes, like those of mammals. For mammalian sequences, accuracy can be improved by automatically removing processed pseudogenes from the predictions using PPFINDER (van Baren and Brent, 2006).

The benefit of N-SCAN's alignment sequence method depends, in part, on the evolutionary divergence between the target and informant genomes. This distance must be large enough so that noncoding regions are less conserved than coding regions, but small enough to find most coding sequences in both species. Based on preliminary data, the authors believe that the method works best for an informant that maximizes the sum of the mismatch and gap percentages in the whole-genome alignment of target and informant. The patterns of mismatches and gaps helps to discriminate between coding and non-coding regions

(e.g., most mismatches in the coding region occur in the third codon position). For a closely related species, the number of mismatches and gaps is small and therefore not very informative. As the evolutionary distance between target and informant increases, the number of mismatches and gaps increases and so does informant utility, until it reaches a maximum. After this point, the sequences are too different to align well over large stretches of the genome, and the percentage of mismatches and gaps decreases (R. Brown and M. Brent, unpub. observ.).

Errors in the target sequence or the assembly can have a major impact on the performance of gene predictions. Particularly bad errors include frame shifts and the introduction or alteration of stop codons or splice sites. For TWINSCAN and N-SCAN, the quality and continuity of the informant database is less important: the accuracy of gene prediction differs only marginally when a database of shotgun reads with 3× coverage is used instead of 7× coverage assembled into contiguous sequences (R. Brown, unpub. observ.).

Overall recommendations

The best gene-prediction method to use depends a great deal on the application. One of the main considerations should be the value of increasing sensitivity, as compared to the cost of decreasing specificity. Sensitivity can often be increased by including predictions by any one of multiple systems, whereas specificity can be increased by considering only predictions made by all systems (Guigo et al., 2003). For high-value projects focusing on a few kilobases of sequence, multiple methods can be used and the results can be combined, either by manual inspection or by specific 'combiner programs' such as JIGSAW (Allen and Salzberg, 2005) and GLEAN (Elsik et al., 2007). However, for high-throughput projects involving megabases of sequence, this may not be feasible. Another consideration is the importance of finding genes that are truly novel. If finding novel genes without sacrificing specificity is important, N-SCAN is the method of choice for most of the genomes described above. If finding novel genes is important but sensitivity is more important than specificity, other gene finders may be better for mammals. Finally, if accurate annotation of genes that are similar to known genes is most important, a method like Ensembl may be best.

N-SCAN versions and new features

N-SCAN is under active development. A Web site for TWINSCAN, the previous implementation, was created in 2003. N-SCAN was published in 2006 and contains considerable improvements. N-SCAN has a more sophisticated evolutionary model than TWINSCAN in that it considers the specific bases in the alignment, rather than treating all mismatches and deletions as identical. It can also use alignments of more than two sequences and model their phylogenetic relationships. N-SCAN predicts 5′ UTRs, including completely non-coding exons, and it can use alignments of expressed sequence tags (ESTs) to improve accuracy. N-SCAN can also be run in TWINSCAN mode. TWINSCAN mode uses a different representation of alignments called conservation sequence, which is composed of three symbols denoting match, mismatch/deletion, or unaligned region, and different parameters are used.

Using N-SCAN to analyze large-scale genomic sequences: Annotations of complete genomes

N-SCAN has been used to annotate all mammalian sequences for which sequence is available at the time of this writing, as well as twelve fruit fly species. TWINSCAN was used for gene prediction on the *Arabidopsis thaliana*, maize, and rice genomes, and on nematode (worm) species. These gene predictions are available through *http://mblab.wustl.edu/predictions* and the UCSC genome browser (*UNIT 1.4*).

Details of the way N-SCAN is run on mammalian and fly genomes and its performance can be found in Gross and Brent (2006). Experimental verification by RT-PCR and sequencing has shown that N-SCAN can identify new genes or additional exons not found in RefSeq (Brown et al., 2005; Wei and Brent, 2006). Additions to N-SCAN are a method for adding EST evidence that greatly improves specificity (Wei and Brent, 2006), and an iterative masking/reprediction method to remove pseudogenes from mammalian gene predictions (van Baren and Brent, 2006). The Publications page of the Brent laboratory Web site (*http://mblab.wustl.edu*) contains references to other papers that have employed TWINSCAN/N-SCAN in their analyses, and whole genome annotations can be downloaded from *http://mblab.wustl.edu/predictions*.

Critical Parameters and Troubleshooting

Parameter sets

N-SCAN relies on a parameter file that contains two parameter sets: target genome parameters and phylogenetic parameters. The target genome parameters describe characteristics of the genome to be analyzed, such as the intron, intergenic, and UTR length distributions, the splice acceptor and donor sites, and the hexamer composition of coding and noncoding sequence. Since genomes from the same clade (e.g., mammals) usually have similar characteristics, it is possible to use parameter sets that were optimized for another species in the same clade. However, it is not advisable to use parameters that were optimized on a more distantly related species, since their genome characteristics may be very different and this will have deleterious effects on accurate gene prediction. The phylogenetic parameters describe the patterns of divergence between two genomes. Accuracy is less sensitive to the phylogenetic parameters than to the target genome parameters. If parameters are not available for a given genome pair, parameters from a pair with similar evolutionary distance and similar target-genome gene-density can be substituted.

The N-SCAN Web site is regularly updated as parameter sets are improved. The Submission page for each N-SCAN run lists the N-SCAN code version, and it also contains a link to the parameter file that was used. This information is essential to reproducing results, so one should be sure to include it in all communications regarding N-SCAN results, both private and published. Users of the Web server cannot choose among parameter sets. Users with local installations can choose input parameter files from the N-SCAN package, or provide their own. To estimate parameters, the program iPEstimate can be downloaded from *http://mblab.wustl.edu/software*. iPEstimate is a versatile parameter estimation program, and comes with detailed information on both parameter estimation theory and use of the program.

For collaboration on adapting N-SCAN to new genomes, E-mail *nscan@mblab.wustl.edu*.

Masking

Masking of simple and interspersed repeats is also a consideration. In general, N-SCAN is both faster and more accurate on sequences that have been aggressively masked for interspersed repeat elements. However, masking of simple and low complexity repeats results in slightly lower sensitivity, because some genuine exons do contain such repeats. The authors have found that N-SCAN is most accurate when interspersed repeats are masked, and low complexity/simple repeats are not, therefore, these are the default settings. However, this may lead to overprediction of repeat-containing exons, and in this case, it is advisable to try another round of prediction with the simple and low complexity repeats masked.

Suggestions for Further Analysis

The N-SCAN group has developed a software package called Eval (Keibler and Brent, 2003) for comparing predictions generated by different programs with each other or with standard annotations. It provides summaries and graphical distributions for many statistics describing any set of annotations, regardless of their source. It also compares sets of predictions to standard annotations and to one another. Eval is open-source software and can be obtained from *http://mblab.wustl.edu/software*.

Acknowledgements

Thanks to Randall Brown for unpublished data on the effects of evolutionary distance on N-SCAN accuracy, to Laura Langton for error-proofing the support protocols, and to Laura Kyro for creating the figures for this unit. This work was supported in part by grants from the National Institutes of Health (HG002278, HG003700), the National Science Foundation (0501758), and Monsanto to Michael R. Brent.

Literature Cited

Alexandersson, M., Cawley, S., and Pachter, L. 2003. SLAM-Cross-species gene finding and alignment with a generalized pair hidden Markov model. *Genome Res.* 13:496-502.

Allen, J.E. and Salzberg, S.L. 2005. JIGSAW: Integration of multiple sources of evidence for gene prediction. *Bioinformatics* 21:3596-3603.

Allen, J.E., Pertea, M., and Salzberg, S.L. 2004. Computational gene prediction using multiple sources of evidence. *Genome Res.* 14:142-148.

Brown, R.H., Gross, S.S., and Brent, M.R. 2005. Begin at the beginning: Predicting genes with 5′ UTRs. *Genome Res.* 15:742-747.

Burge, C. 1997. Identification of Genes in Human Genomic DNA. *In* Stanford Univeristy. Stanford University, Stanford, Calif.

Burge, C. and Karlin, S. 1997. Prediction of complete gene structures in human genomic DNA. *J. Mol. Biol.* 268:78-94.

Elsik, C.G., Mackey, A.J., Reese, J.T., Milshina, N.V., Roos, D.S., and Weinstock, G.M. 2007. Creating a honey bee consensus gene set. *Genome Biol.* 8:R13.

Flicek, P., Keibler, E., Hu, P., Korf, I., and Brent, M.R. 2003. Leveraging the mouse genome for gene prediction in human: From whole-genome shotgun reads to a global synteny map. *Genome Res.* 13:46-54.

Gross, S.S. and Brent, M.R. 2006. Using multiple alignments to improve gene prediction. *J. Comput. Biol.* 13:379-393.

Guigo, R., Agarwal, P., Abril, J.F., Burset, M., and Fickett, J.W. 2000. An assessment of gene prediction accuracy in large DNA sequences. *Genome Res.* 10:1631-1642.

Guigo, R., Dermitzakis, E.T., Agarwal, P., Ponting, C.P., Parra, G., Reymond, A., Abril, J.F., Keibler, E., Lyle, R., Ucla, C., Antonarakis, S.E., and Brent, M.R. 2003. Comparison of mouse and human genomes followed by experimental verification yields an estimated 1,019 additional genes. *Proc. Natl. Acad. Sci. U.S.A.* 100:1140-1145.

Guigo, R., Flicek, P., Abril, J.F., Reymond, A., Lagarde, J., Denoeud, F., Antonarakis, S., Ashburner, M., Bajic, V.B., Birney, E., Castelo, R., Eyras, E., Ucla, C., Gingeras, T.R., Harrow, J., Hubbard, T., Lewis, S.E., and Reese, M.G. 2006. EGASP: The human ENCODE Genome Annotation Assessment Project. *Genome Biol.* 7:S21-S31.

Hubbard, T.J., Aken, B.L., Beal, K., Ballester, B., Caccamo, M., Chen, Y., Clarke, L., Coates, G., Cunningham, F., Cutts, T., Down, T., Dyer, S.C., Fitzgerald, S., Fernandez-Banet, J., Graf, S., Haider, S., Hammond, M., Herrero, J., Holland, R., Howe, K., Howe, K., Johnson, N., Kahari, A., Keefe, D., Kokocinski, F., Kulesha, E., Lawson, D., Longden, I., Melsopp, C., Megy, K., Meidl, P., Ouverdin, B., Parker, A., Prlic, A., Rice, S., Rios, D., Schuster, M., Sealy, I., Severin, J., Slater, G., Smedley, D., Spudich, G., Trevanion, S., Vilella, A., Vogel, J., White, S., Wood, M., Cox, T., Curwen, V., Durbin, R., Fernandez-Suarez, X.M., Flicek, P., Kasprzyk, A., Proctor, G., Searle, S., Smith, J., Ureta-Vidal, A., and Birney, E. 2007. Ensembl 2007. *Nucleic Acids Res.* 35:D610-D617.

Keibler, E. and Brent, M.R. 2003. Eval: A software package for analysis of genome annotations. *BMC Bioinformatics* 4:50.

Korf, I., Flicek, P., Duan, D., and Brent, M.R. 2001. Integrating genomic homology into gene structure prediction. *Bioinformatics* 17:S140-S148.

Mouse Genome Sequencing Consortium et al.. 2002. Initial sequencing and comparative analysis of the mouse genome. *Nature* 420:520-562.

Parra, G., Agarwal, P., Abril, J.F., Wiehe, T., Fickett, J.W., and Guigo, R. 2003. Comparative gene prediction in human and mouse. *Genome Res.* 13:108-117.

Salamov, A.A. and Solovyev, V.V. 2000. Ab initio gene finding in *Drosophila* genomic DNA. *Genome Res.* 10:516-522.

Stanke, M. and Waack, S. 2003. Gene prediction with a hidden Markov model and a new intron submodel. *Bioinformatics* 19:II215-II225.

Stanke, M., Tzvetkova, A., and Morgenstern, B. 2006. AUGUSTUS at EGASP: Using EST, protein and genomic alignments for improved gene prediction in the human genome. *Genome Biol.* 7:S11.1- S11.8.

van Baren, M.J. and Brent, M.R. 2006. Iterative gene prediction and pseudogene removal improves genome annotation. *Genome Res.* 16:678-685.

Wei, C. and Brent, M.R. 2006. Using ESTs to improve the accuracy of de novo gene prediction. *BMC Bioinformatics* 7:327.

GrailEXP and Genome Analysis Pipeline for Genome Annotation

Since its inception, The Gene Recognition and Analysis Internet Link (GRAIL; Uberbacher and Mural, 1991; Mural et al., 1992) has been one of the most widely used systems for locating protein-coding genes and several other features of biological interest in DNA sequences, and has been used extensively for annotation of human and mouse genomes.

This unit first describes the use of GrailEXP (see Basic Protocol), the latest version of this gene-finding system from Oak Ridge National Laboratory (ORNL), where the original GRAIL system was developed. GrailEXP provides significant improvements over GRAIL by exploiting the information gleaned from sequence similarities between the sequence being analyzed and sequences in one or more databases of complete and partial known gene messages, including RefSeq, HTDB, dbEST, EGAD, DOTS, and RIKEN. GrailEXP provides substantially more accurate gene models by making use of sequence similarity with Expressed Sequence Tags (ESTs) and known genes. GrailEXP is also relatively unique in providing alternatively spliced constructs for each gene based on the available EST evidence. GrailEXP has been designed and implemented as a modular system. This facilitates its use as a pure exon finder similar to GRAIL 2, as a sequence alignment program for aligning cDNAs with genomic sequence similar to sim4, or as an expert system for constructing gene models from all available information, namely predicted exons and sequence alignments.

The Alternate Protocol describes the use of the Genome Analysis Pipeline, a Web application which allows users to perform comprehensive sequence analysis by offering a selection from a wide choice of supported gene finders, other biological feature finders, and database searches.

PERFORMING GENE PREDICTIONS USING THE GrailEXP WEB INTERFACE

BASIC PROTOCOL

The GrailEXP Web interface (*http://compbio.ornl.gov/grailexp*) provides the user with selection options for organisms, databases to search, and analysis tasks to perform. The Web interface limits input sequence size to 500,000 bases. Results are returned in the browser window.

Necessary Resources

Hardware

Any computer workstation (PC, Macintosh, Unix, Linux) with Web access

Software

Web browser (e.g., Netscape Navigator, Microsoft Internet Explorer)

Files

DNA sequence of interest in Raw or FASTA format (*APPENDIX 1B*)

1. Open the GrailEXP Web page (*http://compbio.ornl.gov/grailexp*) in the Web browser.

   The GrailEXP Web page will appear in the main browser window (Fig. 4.9.1). The sidebar on the left side of the page provides useful links, including references, the FAQ (Frequently Asked Questions), and license agreement for downloading the software. The main frame on the page displays the analysis request submission form, which provides for user selection of options, and sequence input for the analysis, as outlined in the following steps.

Contributed by Edward C. Uberbacher, Doug Hyatt, and Manesh Shah
Current Protocols in Bioinformatics (2003) 4.9.1-4.9.15
Copyright © 2004 by John Wiley & Sons, Inc.

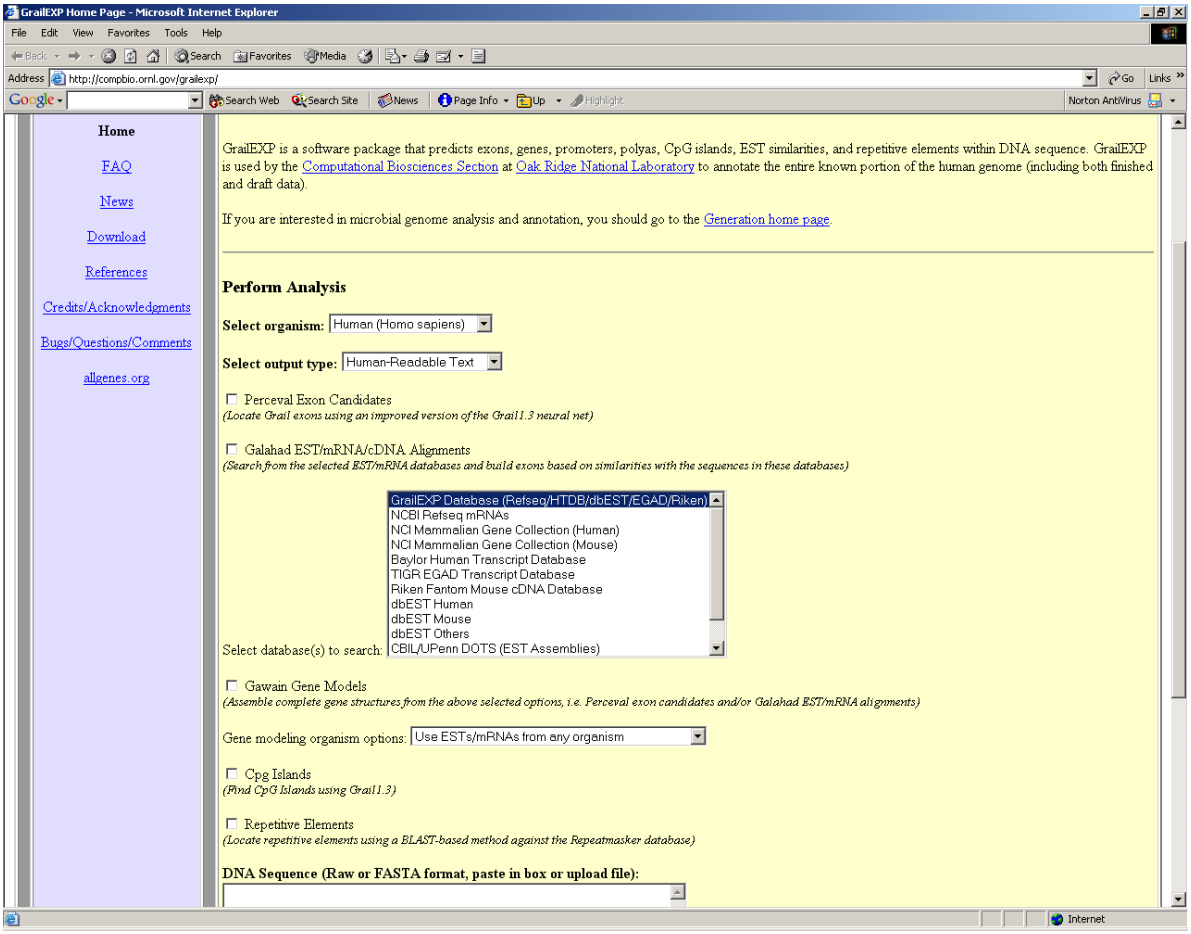

GrailEXP Home Page - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://compbio.ornl.gov/grailexp/
Home
FAQ
News
Download
References
Credits/Acknowledgments
Bugs/Questions/Comments
allgenes.org
GrailEXP is a software package that predicts exons, genes, promoters, polyas, CpG islands, EST similarities, and repetitive elements within DNA sequence. GrailEXP is used by the Computational Biosciences Section at Oak Ridge National Laboratory to annotate the entire known portion of the human genome (including both finished and draft data).
If you are interested in microbial genome analysis and annotation, you should go to the Generation home page.
Perform Analysis
Select organism: Human (Homo sapiens)
Select output type: Human-Readable Text
Perceval Exon Candidates
(Locate Grail exons using an improved version of the Grail1.3 neural net)
Galahad EST/mRNA/cDNA Alignments
(Search from the selected EST/mRNA databases and build exons based on similarities with the sequences in these databases)
Select database(s) to search:
GrailEXP Database (Refseq/HTDB/dbEST/EGAD/Riken)
NCBI Refseq mRNAs
NCI Mammalian Gene Collection (Human)
NCI Mammalian Gene Collection (Mouse)
Baylor Human Transcript Database
TIGR EGAD Transcript Database
Riken Fantom Mouse cDNA Database
dbEST Human
dbEST Mouse
dbEST Others
CBIL/UPenn DOTS (EST Assemblies)
Gawain Gene Models
(Assemble complete gene structures from the above selected options, i.e. Perceval exon candidates and/or Galahad EST/mRNA alignments)
Gene modeling organism options: Use ESTs/mRNAs from any organism
Cpg Islands
(Find CpG Islands using Grail1.3)
Repetitive Elements
(Locate repetitive elements using a BLAST-based method against the Repeatmasker database)
DNA Sequence (Raw or FASTA format, paste in box or upload file):
Internet

Figure 4.9.1 GrailEXP submission form.

Select parameters

2. Select the organism of interest for analysis from the "Select organism" pull-down menu.

   Currently supported organisms include human, mouse, Arabidopsis thaliana, and Drosophila melanogaster. Support for additional organisms is under development.

3. Select the desired output format from the "Select output type" pull-down menu.

   The options for output format are Human-Readable Text, Genome Channel Format, and Raw GrailEXP Format. Human-Readable Text is the only format that is designed for easy visual comprehension, and is the recommended choice. The other two formats are machine-readable formats used by other tools at ORNL. Several other output formats, such as GenBank and GFF, are under development.

4. Check Perceval Exon Candidates checkbox if GRAIL 2 exon prediction option is desired.

   Checking the box labeled Perceval Exon Candidates will run Perceval, an updated version of the GRAIL 2 (neural net-based) exon prediction on the input sequence. This program locates exons using pattern recognition technique only; it does not incorporate any

database or homology information. Users wishing to emulate the behavior of the old GRAIL 1.3 system should select this option. Also see Background Information for discussion of Perceval.

5. Choose whether or not to perform a search of gene message databases and which databases to search.

   Checking the box labeled Galahad EST/mRNA/cDNA Alignments will run Galahad, GrailEXP's BLAST-based alignment algorithm, using the selected databases. By default, the GrailEXP database (which incorporates a number of publicly available databases) is selected, but the user can narrow the search by deselecting this database and selecting one or more component databases. The output from this program will be putative exon boundaries determined from the alignment with the ESTs, mRNAs, and/or cDNAs in the target databases. Also see Background Information for discussion of Galahad.

6. Choose whether or not to assemble the requests from steps 4 and 5 into complete gene structures.

   Checking the box labeled Gawain Gene Predictions will run Gawain, GrailEXP's dynamic programming gene-assembly algorithm, which will assemble complete gene structures from the neural net–predicted exon candidates (if requested in step 4) and the alignment-based exon candidates (if requested in step 5).

   By default, Gawain uses all the alignments obtained from Galahad. However, the "Gene modeling organism options" pull-down menu (located below the Gawain Gene Models check box) allows the user to limit the use of alignments to sequences of only the specified organism. Also see Background Information for discussion of Gawain.

7. Choose whether or not to locate CpG islands.

   Checking the box labeled CpG Islands will locate CpG Islands within the DNA sequence using the Grail 1.3 CpG island-location algorithm.

8. Choose whether or not to locate repetitive elements and mask the sequence for repetitives.

   Checking the box labeled Repetitive Elements will locate simple repeats using the GRAIL 1.3 simple repeat location algorithm. It will also locate complex repeats by performing a BLAST search against the RepeatMasker (A.F.A. Smit and P. Green, unpub. observ.; http://ftp.genome.washington.edu/RM/RepeatMasker.html) database. The sequence corresponding to the located complex repeats will be masked and returned in the browser window.

Input sequence and run analysis

9. Input the DNA sequence. Cut and paste the DNA sequence in the window labeled DNA Sequence or click on the Browse button to select a file on the local computer drives for uploading.

   A valid FASTA format or Raw text sequence must be provided to the program. The sequence must be at least 100 bases in length and no more than 500 kb. It should be emphasized that GenBank or other formats will not work correctly. The sequence may either be cut and pasted into the window or uploaded from a file. Clicking the button labeled Browse (below the DNA Sequence box; not shown in figure) allows the user to search the local disk for the desired file to upload.

10. Click the Go! button (below the DNA Sequence box; not shown in figure) to launch the analysis. Alternatively, click the "Reset form" button to reset the form to its default values and repeat steps 1 to 10.

    Upon launching the analysis, the user will be provided with a summary page showing basic information about the request: the sequence name, the sequence length, a request ID, and an estimated time to complete the analysis.

11. On the page which then appears, click the "Check results" button to check on the progress of the analysis.

    After clicking on the "Check results" button, the user will be redirected to a self-refreshing page which checks every 60 sec for status of the requested tasks. Once the job is complete, the user will be automatically redirected to the results page.

    Many analyses can be quite time-consuming, and the user might not want to sit at the workstation and wait for the results to complete. GrailEXP therefore assigns each request a Request ID so that the user may retrieve the results at a later date. On completion of the analysis, results are returned to a Web page with a URL of the form http://compbio.ornl.gov/GAT_tmp/[req_id].html, where [req_id] represents the Request ID. The user may type this URL into the browser's Address (Internet Explorer) or Location (Netscape) box at any time to check on the progress of the request.

12. Examine the results returned in the browser window.

ALTERNATE PROTOCOL

USING GENOME ANALYSIS PIPELINE FOR COMPREHENSIVE ANALYSIS OF DNA SEQUENCES

Several Genome Analysis Pipelines can be accessed from the ORNL Genome Pipeline Web site (*http://compbio.ornl.gov/genomepipeline*): i.e., Eukaryotic (Human and Mouse), Eukaryotic (Yeast), and Prokaryotic. This discussion focuses on the Eukaryotic (Human and Mouse) pipeline.

The first control on this pipeline Web form (Fig. 4.9.2) is a pull-down menu labeled "Select organism," which allows the user to select the organism of interest. The next control is a check box labeled "Select all services." This control is provided for the convenience of selecting or deselecting all supported analysis services on the form, with a single mouse click. It is also possible to select or deselect the individual supported analysis services on the form. For each selected service, there may be one or more parameter options that can be selected or set.

The Eukaryotic (Human and Mouse) Genome Analysis Pipeline (Fig. 4.9.2) supports several analysis tools. The supported gene finders are GrailEXP and GENESCAN. The user can specify post-processing analysis on the predicted genes. These consist of sequence similarity search, BLASTP (*UNIT 3.4*), and protein family classification analysis (Pfam; *UNIT 2.5*). BLASTP (*UNIT 3.4*) can be run against GenBank nr (nonredundant database) or SwissProt protein databases by selecting one of these options from the Database pull-down menu. Another menu is provided where the user can select BLASTP E-value parameter threshold (*UNIT 3.4*). Additionally, several feature finders are available, consisting of CpG Islands, RepeatMasker, tRNA, BAC-end pairs, and STS e-PCR. The user can also select the option to run BLASTN (*UNIT 3.3*) on the input DNA sequence against one of several DNA databases.

Once the user has selected the analysis tools of interest and has set relevant options, the sequence of interest can be loaded, either by cutting and pasting the sequence into the DNA Sequence text box, or by using the Browse button to select a sequence file from the local file system. If the user simply wishes to perform a demo run, the Demo checkbox can be checked instead of loading a sequence. The request can then be submitted by clicking on the Submit Request button. For the purposes of this example, the same data set used in the Basic Protocol (`humadag`) will be used here.

On submission of the request, the server returns a request status page, which periodically refreshes itself to provide the user with an indication regarding the precise status of pipeline processing, and failure or completion of each analysis component. On completion of the entire pipeline processing, the final pipeline status page is displayed with a button labeled Get Summary. On clicking this button, the server returns a summary page (Fig. 4.9.3) with a page similar to the status page, but with additional information about the number of hits found by each analysis component, and hyperlinks to the individual analysis results, accessible in several

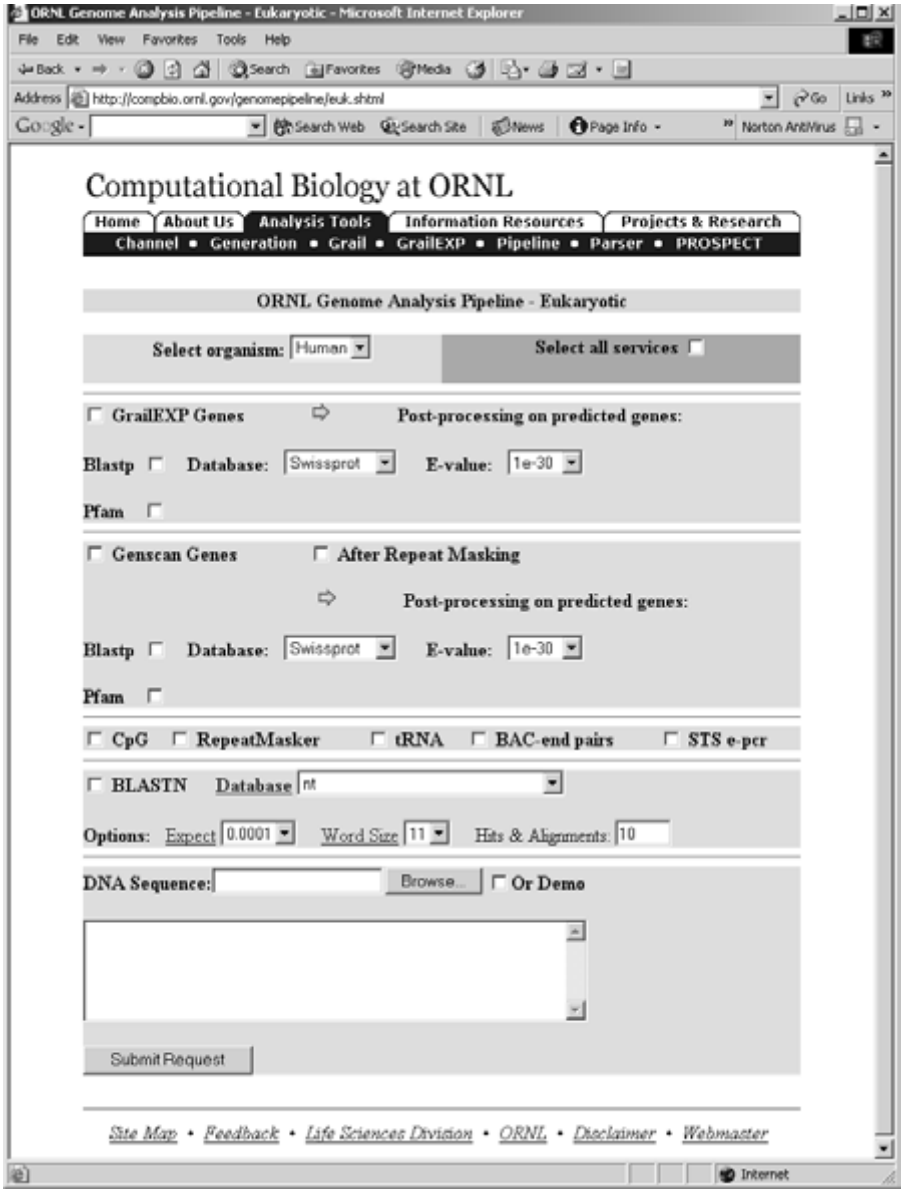

ORNL Genome Analysis Pipeline - Eukaryotic - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://compbio.ornl.gov/genomepipeline/euk.shtml
Computational Biology at ORNL
Home About Us Analysis Tools Information Resources Projects & Research
Channel • Generation • Grail • GrailEXP • Pipeline • Parser • PROSPECT
ORNL Genome Analysis Pipeline - Eukaryotic
Select organism: Human
Select all services
GrailEXP Genes
Post-processing on predicted genes:
Blastp Database: Swissprot E-value: 1e-30
Pfam
Genscan Genes
After Repeat Masking
Post-processing on predicted genes:
Blastp Database: Swissprot E-value: 1e-30
Pfam
CpG RepeatMasker tRNA BAC-end pairs STS e-pcr
BLASTN Database nt
Options: Expect 0.0001 Word Size 11 Hits & Alignments: 10
DNA Sequence: Browse... Or Demo
Submit Request
Site Map • Feedback • Life Sciences Division • ORNL • Disclaimer • Webmaster
Internet

Figure 4.9.2 Pipeline submission form.

different formats. When an analysis module fails, instead of the green Succeeded icon, the user would see a red Failed icon for that module. For subsequent analysis modules that are dependent on the failed analysis module, the user would see a red Aborted icon. Subsequent analysis modules that are independent of the failed module would be run normally.

The Summary page also provides three useful links for retrieval and display of the pipeline results. The Raw output link allows the user to retrieve the pipeline results as a single text document. Individual analysis output can be viewed in Raw or Text Table format by selecting the appropriate menu option from the pull-down menu provided next to the Status icon for that particular analysis (refer to Fig. 4.9.3). Figure 4.9.4 illustrates the Text Table output for GrailEXP Gene Finder analysis. It consists of a summarized list of genes, followed by detailed information about each predicted gene and its components [consisting of exons, promoters and poly(A) sites]. This is followed by the list of protein and mRNA sequences for each of the predicted GrailEXP genes.

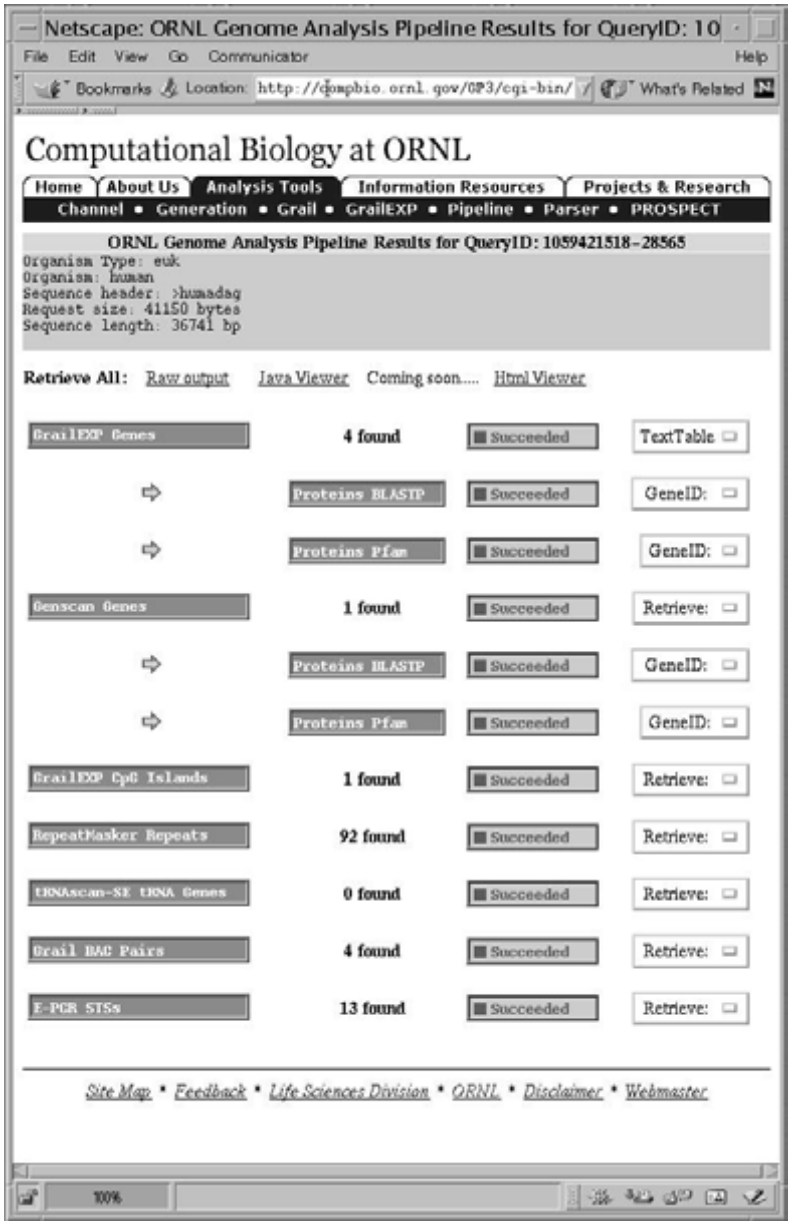

Netscape: ORNL Genome Analysis Pipeline Results for QueryID: 10
File Edit View Go Communicator Help
Bookmarks Location: http://compbio.ornl.gov/GP3/cgi-bin/ What's Related
Computational Biology at ORNL
Home About Us Analysis Tools Information Resources Projects & Research
Channel • Generation • Grail • GrailEXP • Pipeline • Parser • PROSPECT
ORNL Genome Analysis Pipeline Results for QueryID: 1059421518-28565
Organism Type: euk
Organism: human
Sequence header: >humadaq
Request size: 41150 bytes
Sequence length: 36741 bp
Retrieve All: Raw output Java Viewer Coming soon.... Html Viewer
GrailEXP Genes 4 found Succeeded TextTable
Proteins BLASTP Succeeded GeneID:
Proteins Pfam Succeeded GeneID:
Genscan Genes 1 found Succeeded Retrieve:
Proteins BLASTP Succeeded GeneID:
Proteins Pfam Succeeded GeneID:
GrailEXP CpG Islands 1 found Succeeded Retrieve:
RepeatMasker Repeats 92 found Succeeded Retrieve:
tRNAscan-SE tRNA Genes 0 found Succeeded Retrieve:
Grail BAC Pairs 4 found Succeeded Retrieve:
E-PCR STSs 13 found Succeeded Retrieve:
Site Map • Feedback • Life Sciences Division • ORNL • Disclaimer • Webmaster
100%

Figure 4.9.3 Pipeline summary page.

The Java Viewer link allows the user to download a Java (Swing) applet for graphical display and visualization of the pipeline results. The applet (Fig. 4.9.5) consists of three sections: (1) Feature Display, which displays genes and other identified biological features in a graphical form; the horizontal gray scale bar in the center of the top Feature Display window represents the local GC content of the input DNA sequence, with brightness proportional to the percentage GC content of that region of the sequence; (2) Sequence Display, which displays a 100-base (double-stranded) region corresponding to the stretch of DNA selected using the scroll bar at the bottom of the Sequence Display; and (3) Features Pane, which is a tabbed pane with a table for each of the analysis features. Each table displays the list of feature elements for that feature type, and their locations, scores, and other relevant information.

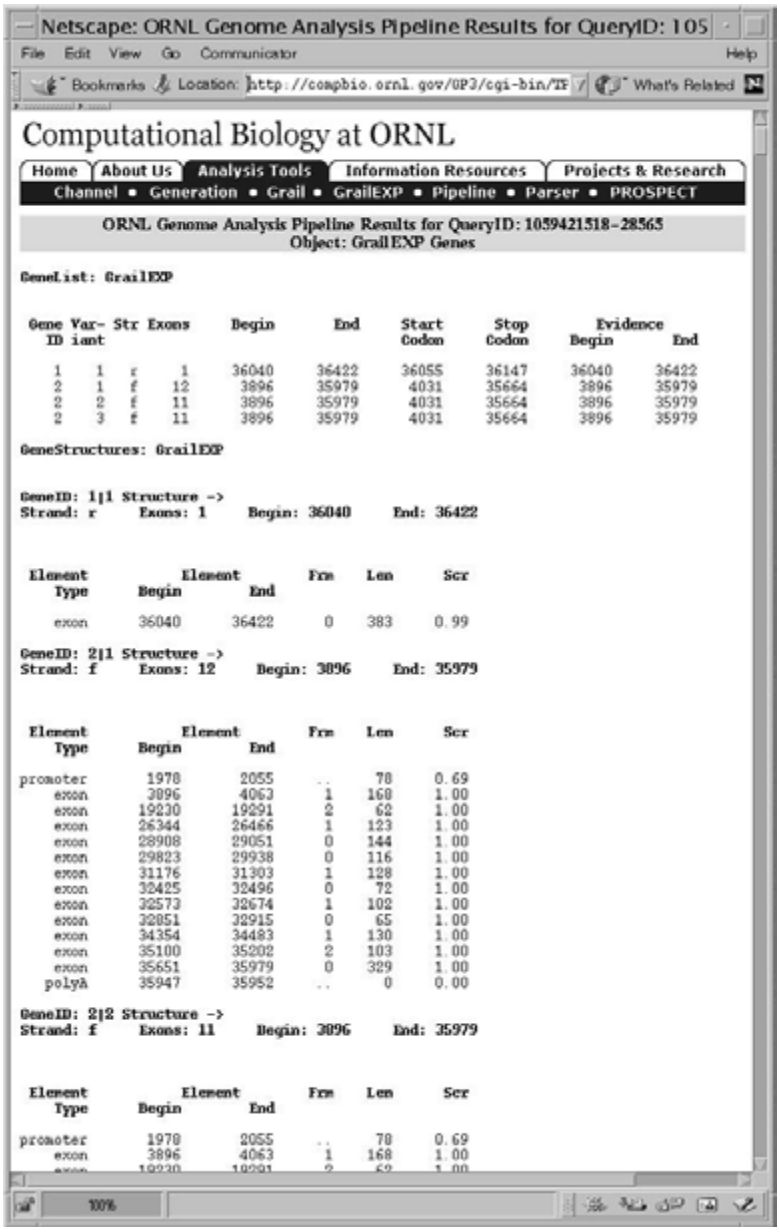

Netscape: ORNL Genome Analysis Pipeline Results for QueryID: 105
File Edit View Go Communicator Help
Bookmarks Location: http://compbio.ornl.gov/GP3/cgi-bin/TF What's Related
Computational Biology at ORNL
Home About Us Analysis Tools Information Resources Projects & Research
Channel • Generation • Grail • GrailEXP • Pipeline • Parser • PROSPECT
ORNL Genome Analysis Pipeline Results for QueryID: 1059421518-28565
Object: GrailEXP Genes
GeneList: GrailEXP
Gene ID Var-iant Str Exons Begin End Start Codon Stop Codon Evidence Begin Evidence End
1 1 r 1 36040 36422 36055 36147 36040 36422
2 1 f 12 3896 35979 4031 35664 3896 35979
2 2 f 11 3896 35979 4031 35664 3896 35979
2 3 f 11 3896 35979 4031 35664 3896 35979
GeneStructures: GrailEXP
GeneID: 1|1 Structure ->
Strand: r Exons: 1 Begin: 36040 End: 36422
Element Type Element Begin Element End Frm Len Scr
exon 36040 36422 0 383 0.99
GeneID: 2|1 Structure ->
Strand: f Exons: 12 Begin: 3896 End: 35979
Element Type Element Begin Element End Frm Len Scr
promoter 1978 2055 .. 78 0.69
exon 3896 4063 1 168 1.00
exon 19230 19291 2 62 1.00
exon 26344 26466 1 123 1.00
exon 28908 29051 0 144 1.00
exon 29823 29938 0 116 1.00
exon 31176 31303 1 128 1.00
exon 32425 32496 0 72 1.00
exon 32573 32674 1 102 1.00
exon 32851 32915 0 65 1.00
exon 34354 34483 1 130 1.00
exon 35100 35202 2 103 1.00
exon 35651 35979 0 329 1.00
polyA 35947 35952 .. 0 0.00
GeneID: 2|2 Structure ->
Strand: f Exons: 11 Begin: 3896 End: 35979
Element Type Element Begin Element End Frm Len Scr
promoter 1978 2055 .. 78 0.69
exon 3896 4063 1 168 1.00
100%

Figure 4.9.4 GrailEXP genes text table.

The Java Viewer applet uses Sun's Java 2 plugin. If the user does not have the plugin installed, upon loading the Java Viewer page, the browser will automatically point to Sun's Java Plugin download page and prompt the user for permission to download and install the plugin. If the user agrees, automatic plugin download will be initiated. If there is a problem with automatic installation, the user can manually download the plugin from Sun's Web site at *http://java.sun.com/getjava/download.html*, and install it. The Java Viewer is under active development at the time of this writing. New functionality is being added and therefore the user may have access to more features than are covered above. Since Java itself is undergoing rapid development, the Java Viewer may be modified to take advantage of newer features. Therefore it may be necessary to install a new version of the plugin. The pipeline Web page will be updated to provide information about the current and expected enhancements, as well as User's Guide and Help sections.

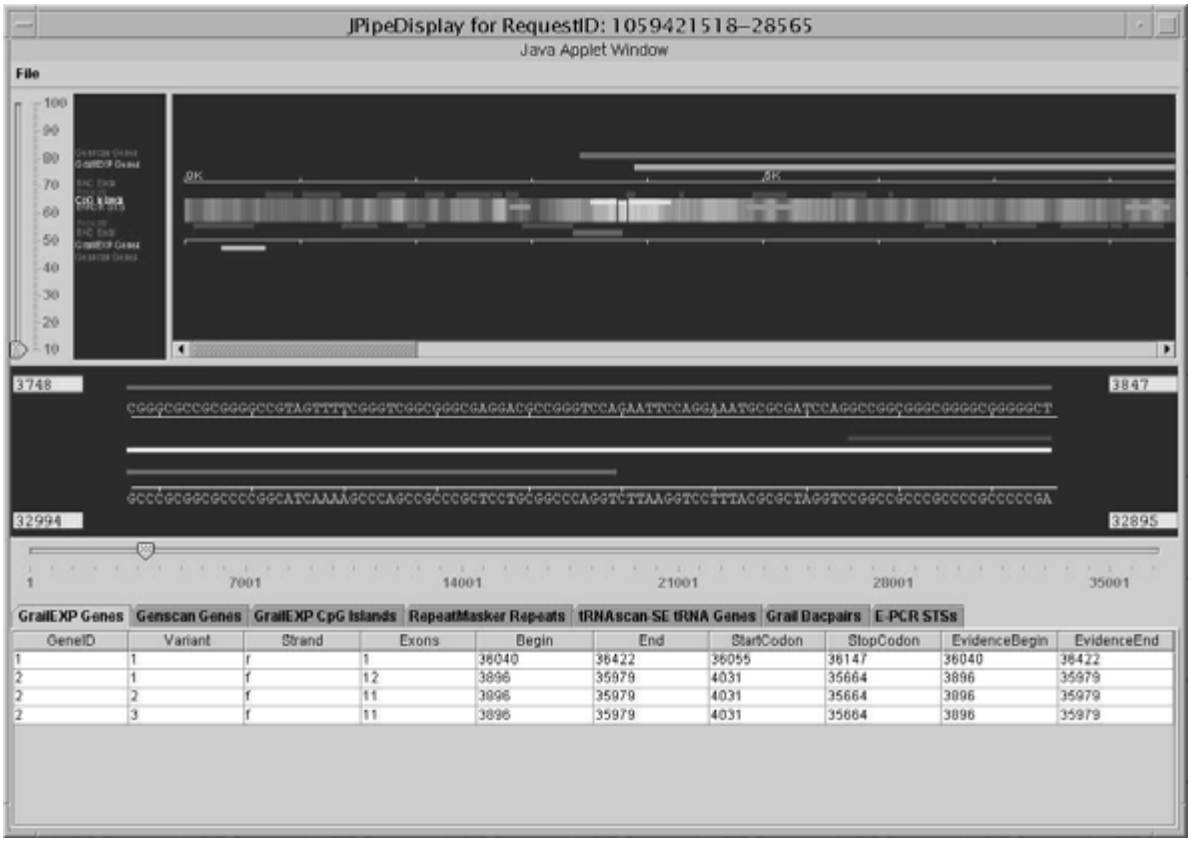


Figure 4.9.5 Java Pipeline Viewer. *This black and white facsimile of the figure is intended only as a placeholder; for full-color version of figure go to http://www.interscience.wiley.com/c_p/colorfigures.htm.*

GUIDELINES FOR UNDERSTANDING RESULTS

Major Gene Modeling Issues in GrailEXP (Gawain)

Alternative splicing recognition

Gawain currently looks for a very specific case of alternative splicing. It looks for fully determined EST evidence that indicates the insertion or deletion of an exon. In other words, if one gene model contains exons A, B, and C, and another contains only exons A and C, then the program identifies this case as an alternative splice. Except in the case of repeating gene regions (which can be incorrectly reported as regions with a lot of alternative splices), Gawain identifies inserted/omitted exons with absolute certainty. However, it cannot currently identify other types of less obvious alternative splicing. It is not a trivial matter to distinguish between an "unspliced" product that has slipped into the EST database, an error in the genomic alignment, and a genuine case of alternative splicing. Just because one rogue EST disagrees with the other evidence does not necessarily mean it is an alternative splice. Even if multiple ESTs indicate an alternative splicing, it could simply be due to an error in the alignment or in the ESTs. In future versions, identification of additional types of alternative splicing will be addressed.

Identification of 5′ and 3′ untranslated regions

The primary indicator of 5′ and 3′ untranslated regions is an additional stop codon in the mRNA. In such cases, the program attempts to find the 5′ and 3′ untranslated region boundaries. The mRNA is analyzed to find the highest-scoring run without a stop codon (the score being based on the length of the run and the GRAIL coding score). It then examines that "run" to find the highest-scoring start codon (based on proximity to the beginning of the run, which exon it falls in, GRAIL start site score, and coding/noncoding scores around the boundary). Gawain correctly identifies the start site in a complete mRNA about 95% of the time. Further performance improvement in future versions will be addressed by adding protein similarity search to the system.

Identification of pseudogenes

GrailEXP does not specifically label a pseudogene as such. However, it does provide some indication of a particular model being a pseudogene. A hyphen ("-") in the protein translation of a gene model indicates a frame shift between exons. One or more of these may indicate a pseudogene. Additionally, if Gawain reports a long mRNA but a short protein translation, this may indicate a pseudogene containing several stop codons; however, this may also indicate an error in the genomic sequence. It is not easy to distinguish between actual pseudogenes and errors in the alignment program due to a missed short exon which might cause a frame shift, or a wrongly identified stop codon due to sequencing error, which confuses the 5′/3′ UTR finder.

Open- and closed-end based gene modeling

Gawain can be run in open or closed end gene modeling mode. Open-ended gene modeling allows "partial" genes to be predicted at the ends, i.e., it assigns no penalties for genes running off the edges of the sequence. This is the ideal way to run the program for contigs known not to begin or end a region. Running with closed ends, on the other hand, will penalize partial genes at the ends of the sequence and will try to "close off" all gene models. This is ideal if one knows that a clone only contains one gene that one wants to examine, and one does not want Gawain to predict partial genes on the ends of the sequence. Gawain runs with closed ends by default, since, typically, there is almost always an initial or terminal GRAIL Exon Candidate that can be predicted near the beginning or end of a sequence. In open-ended mode, it is common to find single-exon partial genes predicted near the edges of the query sequence.

Single-strand gene modeling

By default, Gawain runs in double-stranded gene-modeling mode. However, this can cause problems with recognition of embedded genes, which may be omitted, or the embedding gene may be broken. Gawain can be run on just the forward strand or reverse strand. In fact, the most accurate way to perform gene modeling with Gawain would be to predict exons/alignments in double-stranded mode, then run the gene assembly program twice—once on the forward strand, once on the reverse strand.

ESTs and cDNAs not listed as gene evidence

If an EST/cDNA is not listed as gene evidence, but it is in the same location as the gene model, then there is something about that EST/cDNA that made it inconsistent with the gene model. Either it did not align to the same splice sites or it trickled over the edges into a predicted intron. This happens in instances where an EST's first base falls very near a splice-site edge; often there are not enough bases to align with the next exon upstream, so the program winds up extending the alignment into an intron. This problem can be solved by an examination of the alignments in the gene-modeling phase and subsequent correction of such cases. EST/cDNAs that are inconsistent with gene models are flagged, but are not currently returned to the user. If deemed useful, a future version of Gawain

could be modified to cluster these EST/cDNAs together using GrailEXP's consistency-check function. Since the primarily goal of Gawain is gene modeling, and not EST/cDNA clustering, its current emphasis is on clustering together the EST/cDNAs that support a particular gene model and ignoring the rest.

COMMENTARY

Background Information

Gene-finding programs

Perhaps the first and most fundamental question one might ask of a DNA sequence is whether it is likely to encode the sequence of a protein. Several patterns exist in protein-coding DNA sequences that are not found in noncoding regions. These allow methods to be developed for identifying such regions. Early attempts to distinguish coding from noncoding DNA relied upon base composition, the triplet nature of genetic code, and for some organisms, the frequency of codon usage (Fickett, 1982; Staden, 1984; Fickett and Tung, 1992). Such techniques work reasonably well on long (>200-base) coding regions such as those found in prokaryotes. They are, however, inadequate for predicting the protein-coding regions found in higher eukaryotes, which are relatively short (~130-base) sequences separated by introns that may be hundreds or thousands of bases long. A number of protocols have been devised to identify genes—or the protein-coding portions of genes—by identifying specific functional sites such as splice-donor and splice-acceptor junctions (Brunak et al., 1990, 1992; Guigo et al., 1992; Hutchinson and Hayden, 1992). Techniques have also been devised to locate coding regions by their statistical (Claverie et al., 1990) or periodic properties (Mani, 1992). In addition, there are gene-finding techniques that combine these types of information in various ways (Gelfand, 1990; Uberbacher and Mural, 1991; Mural et al., 1992; Snyder and Stormo, 1993; Xu et al., 1994a).

The first GRAIL system (GRAIL 1) evaluated seven statistical parameters distinguishing coding from noncoding regions, for a window of 100 bases of sequence. The values for each window were then processed by an artificial neural network that had been trained to distinguish coding from noncoding regions on the basis of these parameters. This system performed quite well—it was able to recognize 90% of annotated protein-coding regions ≥100 bases in length from a test set of 19 genes (Uberbacher and Mural, 1991). An acceptably low false-positive rate—approximately one in five predictions when evaluated for both strands—was obtained. Though GRAIL 1 was very useful and widely used, it left several problems unsolved. The fixed window size (100 bases) made it difficult for the system to recognize exons <100 bases in length. Furthermore, the system did not define the edges (splice-acceptor and splice-donor sites) of the exons. To address these problems, GRAIL 2 was developed (Xu et al., 1994a). Instead of examining the coding potential of a segment of sequence of fixed length, GRAIL 2 considers all segments of sequence between minimal splice junctions (YAG for acceptors and GT for donors). Though this initially presents a large number of candidates to the system, that number is quickly reduced by applying a series of rules—e.g., the requirement for at least one open reading frame per exon. Those candidates that remain are screened by an artificial neural network that has been trained using a set of eleven parameters related to exon recognition; this further reduces the set of potential candidates. After the candidates are assigned to clusters, the "best" candidate for each coding exon is selected.

GrailEXP uses the improved GRAIL 2 exon prediction system, detecting 91% of all coding exons—regardless of size—with a false-positive rate of ~9%. In addition, the system correctly predicts both edges for 61% of predicted exons and at least one correct edge for 96% of predicted exons.

Clearly, locating coding regions in DNA sequence is critical to the interpretation of genomic DNA sequence. Besides GRAIL, a number of other approaches have been applied to the problem. geneid (Guigo et al., 1992; *UNIT 4.3*) is a system that combines various rules and statistical features in an attempt to model spliced messenger RNAs. Other attempts to identify coding regions have used dynamic programming and neural networks (Geneparser; Snyder and Stormo, 1993), oligonucleotide composition and discriminant analysis (Solovyev et al., 1994), and linguistic methods (Genlang; Dong and Searles, 1994).

DNA dialects vary among different organisms. Parameters such as the frequency with

which individual codons are used, the length of introns and exons, and the amount of repetitive DNA differ for different organisms. The GrailEXP system has been optimized for human DNA, though it appears to function reasonably well for most mammalian species. As more distantly related organisms are studied, it is necessary to construct specialized systems for each organism. GrailEXP now supports the analysis of sequences from *Drosophila melanogaster*, *Arabidopsis thaliana*, and the mouse.

If the coding exon–recognition portion of the GRAIL 2 system performed perfectly, assembling a model of the spliced mRNA (i.e., the gene model) would be trivial—involving nothing more than connecting the ends of the predicted exons. However, because the coding-region prediction is less than perfect, a computational method must be used to test various combinations of exons in order to propose a gene model. The gene-assembly program (Gawain) in GrailEXP uses dynamic programming to assemble gene models from the predicted exons (Xu et al., 1994b). There are a number of constraints placed on the gene model based on splicing considerations. For example, not only must an open reading frame be maintained over both exons when two predicted exons are connected, but the reading frame(s) that have been predicted for each of the exons must be maintained. These sorts of constraints can be used to evaluate the model-building process at each step, and to force the system to explore other alternatives if there are violations.

Description of GrailEXP system

GrailEXP comprises three major components (Hyatt and Uberbacher, 2002): (1) Perceval, which provides exon prediction as well as CpG Island and Repetitive element prediction, (2) Galahad, which provides gene message alignment functionality, and (3) Gawain, which performs gene assembly.

Perceval: Perceval (Protein-coding Exon, Repetitive, and CpG Island EVALuator) reads in a DNA sequence and produces a list of possible GRAIL Exon Candidates. It provides user options for locating repetitive elements and CpG Islands, and for filtering the exon candidates against a repetitive element database.

A GRAIL Exon Candidate is a region of the sequence, identified by the GRAIL neural network as being a potential exon on the forward or reverse complementary DNA sequence, with a start codon or an AG acceptor splice site at its starting position and a stop codon or a donor splice site at its ending position, all in the same reading frame. The exon candidates are categorized into clusters, with the highest-scoring exon in each cluster clearly indicated. These "best" exons in each cluster are traditionally referred to as "GRAIL exons," rather than candidates.

Perceval identifies and scores all potential splice sites within the sequence, using neural networks trained for recognizing start codons, stop codons, AAG and YAG acceptors, and GT donors, respectively. Low-scoring splice sites are discarded. All possible exon candidates that can be constructed with the remaining splice sites are then scored for their coding potential. Here too, the low-scoring candidates are discarded. The remaining exon candidates are evaluated by another neural network, which is fed the splice site and coding potential scores, as well as the GC content. Once again, only the high-scoring exon candidates are retained as the final set.

This set of exon candidates is then organized into clusters of overlapping exons. Each cluster is filtered for repetitive elements, using NCBI's BLAST program (*UNITS 3.3 & 3.4*). Non-exonic regions of the sequence are substituted with N's and a BLAST search of this sequence is run against a database of repetitive elements. If a repetitive element is determined to have a significant overlap (10% of the exon if overlapping an edge, 50% if embedded inside the exon) with an exon candidate, then that exon candidate is flagged as repetitive element and marked for elimination. Next, a strand-resolution process is applied wherein overlapping exons on opposite strands are examined and the lower-scoring cluster (containing what the authors call "shadow exons") is marked for elimination. The final list of exon candidates is then provided in the output, with the shadow and repetitive exons appropriately flagged. A sequence can also be masked for repetitive elements prior to submission to GrailEXP's exon-prediction program, but this is not recommended as it may lead to the loss of legitimate exons.

Galahad: Galahad (Gene message ALignment) has three major usage modes:

1. Read in a sequence and a GRAIL Exon Candidate file, search a database of partial/complete gene messages, and produce a list of gene message alignments.

2. Read in a sequence file only, search the entire sequence against a database of partial/complete gene messages, and produce a list of gene message alignments.

3. Read in a sequence file and an accession number of a partial/complete gene message and produce an alignment of the two sequences.

A **gene message alignment** is an alignment between gene components (exons) in a genomic sequence and a spliced gene message. It differs from a regular global alignment because there is additional information that can help with the alignment process, most notably that the internal "edges" of each alignment piece should fall on standard splice site (GT....AG) boundaries. A gene message alignment program aligns a gene message with a genomic sequence such that the exon/intron boundaries are clearly identified.

The databases currently used by Galahad at ORNL are CBIL/Penn DOTS Database, TIGR EGAD Transcript Database, NCBI RefSeq Database, NCBI dbEST Database, and a curated set of mRNAs from Genbank.

The default input to Galahad is a DNA query sequence and a list of GRAIL Exon Candidates. It replaces all non-exonic portions of the query sequence with N's, and runs a BLAST search of this sequence against the search database. Only the high-scoring BLAST alignments are retained, thus eliminating a significant number of poor alignments. A temporary database is created from the retained ESTs/cDNAs, and a BLAST search (*UNIT 3.3*) of the entire sequence is run against this database. These raw BLAST alignments are arranged into valid gene structures by applying a dynamic programming algorithm. A merge check algorithm locates the fragments that BLAST failed to join and merges these alignments. An exhaustive search of all splice sites in the vicinity of each edge is performed in order to select the optimal set of left, right, and internal edges. A donor splice site scoring algorithm is used to pick the most likely donor splice site. The resulting alignments, each consisting of exons, introns, and splice sites, are the final gene message alignments.

The program has an effective mechanism to deal with repetitive DNA elements. During the initial BLAST search (*UNIT 3.3*), the program looks for the word "repetitive" in the FASTA header *APPENDIX 1B* of the sequence that is hit. It also identifies clusters of single alignments in the same location in the sequence that hit portions of many ESTs. If ESTs in this cluster are labeled "repetitive," it eliminates those alignments. No EST is ever eliminated that hits in two or more places in what looks like a potential gene. In addition, any single-fragment alignments that do not cover at least 50% of that EST are eliminated.

The gene message database can often be quite large (as of the time of this writing it is 10.9 Gb). In order to perform the database-alignment phase with limited resources and to speed up the process, Galahad has been designed to run in serial or parallel mode using a multiple partition version of the database. In serial mode, each partition is searched sequentially. In parallel mode, all partitions are searched concurrently on different machines. There are several advantages to the multiple-partition scheme over running BLAST multithreaded against a huge database. The databases are loaded into memory in parallel. Each compute node uses substantially less memory. The BLAST searches (*UNITS 3.3 & 3.4*) are less likely to hang when searching smaller databases. The Galahad alignment phase is run in parallel. In either mode, BLAST can of course run in its multithreaded mode. The parallel search is accomplished through the use of TCP/IP client-server system (using Daniel Bernstein's ucspi-tcp program), thus avoiding the overhead of PVM or MPI implementation.

The use of BLAST for obtaining initial approximate alignments, rather than the Smith-Waterman-like algorithm used by some other systems, provides immense speedup. GrailEXP refines these alignments in a subsequent phase. A drawback of using BLAST is that it does not look for short alignments. This means that finding short internal exons must be done by the Galahad alignment program, which is currently being refined for this purpose. The BLAST simple-repeat filtering option causes a break in otherwise good alignments. Ideally, BLAST should filter for simple repeats initially, but then "de-filter" to join fragments together that it incorrectly split. Galahad attempts to join together such breaks, provided the length of the break is less than 200 bases.

Galahad uses dynamic programming to pick the "best" alignments, such that those with the highest percent identity get joined together. This causes incorrect alignments to be reported in the case of multi-exon repeating genes, such as back-to-back-to-back zinc fingers, in close proximity to each other. The first exon of gene 1 will often be joined with the second exon of gene 2, and so on. A future improvement being considered is a module for recognition of repeat gene regions, which would identify and suitably handle such occurrences.

Galahad can identify exons as short as 10 bases, by identifying suitable splice sites to which to align, while processing a much longer BLAST alignment. GrailEXP finds short exons

(as short as 5 bases) at the edges of the alignments, provided the bases in the message exactly match the genomic sequence. This is critical for aligning the CDS portions of transcripts with genomic sequence, where there could be very short exons at the edges of the gene. A future improvement being considered is robust recognition of short exons in which the possible presence of such exons is first determined by detecting a "bump" in the alignment and a high donor splice site score. The current implementation does place the alignment pieces in proper frames, so missing a short exon will only cause a slight ripple in the resulting translation.

It is possible to use exon predictions from another exon or gene-finding system, like GENESCAN, as input to Galahad. In fact, Galahad can interpret GENESCAN output files. Additionally, exons from any other program can be used, by formatting the exon information into the GrailEXP exon format or GENESCAN output format. Additionally, it is not mandatory to provide an exon candidate input file to Galahad. In absence of exon input, Galahad performs alignment of the entire sequence with the search database. In such cases, it is recommended that a repeat-masked sequence be provided as input; otherwise, the processing time would be significant due to the large number of alignments that would be identified. Overall, Galahad provides the advantage of speed and flexibility compared to other alignment systems.

Gawain: Gawain (Gene Assemblies with Alignment Information), the GrailEXP gene-assembly program, builds gene models based on GRAIL Exon Candidates and/or GrailEXP genomic alignments. A GrailEXP gene model incorporates the following components: a 5′ untranslated region, a coding region, a 3′ untranslated region, exons, introns, a polyadenylation [poly(A)] site, a promoter, and all reference sequence evidence consistent with the gene model. The program also generates the mRNA and protein translation for each gene model. New capabilities in this version of GrailEXP include recognition of alternative splicing and clustering of all ESTs/cDNAs that support a particular gene model. Gawain can construct gene models from either exon candidates or alignments, or both.

Gawain first clusters the genomic alignments, and each alignment is assigned a frame using a recursive frame-scoring function that considers the GRAIL coding score for each potential frame, the frames of GRAIL exons that match the alignments, and "spliceability" among the various alignments. An elaborate dynamic programming algorithm is then used to construct gene models. Each node in the dynamic programming model is a GRAIL exon candidate or an alignment-based exon. Connections between the various nodes are scored, with bonus for "good" connections and penalties for "bad" connections. Once the highest-scoring dynamic programming model has been calculated, the preliminary gene models are refined. In this phase, genes can be split or merged, and the exon edges are tweaked to better match splice sites. Alternative splices are then identified and added to the gene table. In the next phase, potential coding starts and stops within each mRNA are identified and evaluated to determine the correct one. In the absence of protein similarity evidence, this is not a perfect process. In addition, false stop codons in the genomic sequence (because of sequencing errors or pseudogenes) can result in incorrect identification of start and stop codons.

Finally, poly(A) sites and promoters are located. Poly(A) recognition uses a simple Markov model. The promoter system uses a neural net and looks for specific types of signals (TATA, CAAT, GC-box). Not every gene model will have a poly(A) site or a promoter. The poly(A) site recognizer scans the sequence for the pattern AATAAA. It then examines a 72-base region around the sequence using a simple Markov model and reports back a score for that site. If a poly(A) site is within 5000 bases of the stop codon of a gene and does not fall in an intron, then it is retained. At most, one poly(A) site (the highest-scoring, in the case of multiples) is assigned to each gene model. The promoter recognition system looks for a TATA or ATA and examines the region around these bases. In particular, the neural net is fed information on GC content, information on CAAT, GGGCGG, and ATG patterns located in close proximity, and sequence fragments flanking these patterns. The neural net evaluates the scores and assigns a total score to the promoter. Again, the promoter must be within 5000 bases of a start codon of a predicted GrailEXP gene model to be retained and cannot fall within an intron. At most, one promoter element is assigned to each gene model.

Alternate methods for accessing GrailEXP

The Genome Channel: Genome Channel (*http://compbio.ornl.gov/channel*) is a Web-based annotation system comprising GrailEXP gene predictions for a variety of organisms, including human, mouse, and many microbial

genomes. Genome Channel is the recommended starting point for users interested in viewing precomputed gene models. A description of Genome Channel is beyond the scope of this unit; users are referred to the above Web site.

Command line: GrailEXP is available for download (*http://compbio.ornl.gov/grailexp/gxpfaq.html*) by academic and nonprofit institutions, allowing them to use the command-line version, which provides the most powerful access to all the options and commands associated with GrailEXP. A detailed description of the possible options and usages for the command-line GrailEXP may be found in the GrailEXP Frequently Asked Questions (FAQ) Web page, mentioned above. The requirements for the command-line version of GrailEXP include a Unix (Linux, Alpha, Solaris, Silicon Graphics) workstation with at least 128 Mb RAM, running Perl 5.0 or higher. GrailEXP analysis involves BLAST searches against several databases. This necessitates downloading and formatting the desired databases for use with GrailEXP.

Literature Cited

Brunak, S., Englebrecht, J., and Knudsen, S. 1990. Neural network detects errors in the assignment of mRNA splice sites. *Nucl. Acids Res.* 18:4797-4801.

Brunak, S., Englebrecht, J., and Knudsen, S. 1992. Prediction of human mRNA donor and acceptor sites from the DNA sequence. *J. Mol. Biol.* 220:49-65.

Claverie, J.-M., Sauvaget, I., and Bougueleret, L. 1990. K-tuple frequency analysis: From intron/exon discrimination to T-cell epitope mapping. *Methods Enzymol.* 183:237-252.

Dong, S. and Searles, D.B. 1994. Gene structure prediction by linguistic methods. *Genomics* 23:540-551.

Fickett, J.W. 1982. Recognition of protein coding regions in DNA sequences. *Nucl. Acids Res.* 10:5303-5318.

Fickett, J.W. and Tung, C.-S. 1992. Assessment of protein coding measures. *Nucl. Acids Res.* 20:6441-6450.

Gelfand, M.S. 1990. Computer prediction of the exon-intron structure of mammalian pre-mRNAs. *Nucl. Acids Res.* 18:5865-5869.

Guigo, R., Knudsen, S., Drake, N., and Smith, T. 1992. Prediction of gene structure. *J. Mol. Biol.* 226:141-157.

Henikoff, S. and Henikoff, J. 1991. Automated assembly of protein blocks for database searching. *Nucl. Acids Res.* 19:6565-6572.

Hutchinson, G.B. and Hayden, M.R. 1992. The prediction of exons through an analysis of spliceable open reading frames. *Nucl. Acids Res.* 20:3453-3462.

Hyatt, D. and Uberbacher, E.C. 2002. Computational DNA sequence analysis and annotation. *In* Genomic Technologies: Present and Future (D.J. Galas and S.J. McCormack, eds.) pp. 345-374. Caister Academic Press, Norfolk, U.K.

Mani, G.S. 1992. Long-range correlations in DNA and the coding regions. *J. Theor. Biol.* 158:447-464.

Mural, R.J., Einstein, J.R., Guan, X., Mann, R.C., and Uberbacher, E.C. 1992. An artificial intelligence approach to DNA sequence feature recognition. *Trends Biotech.* 10:67-69.

Snyder, E.E. and Stormo, G.D. 1993. Identification of coding regions in genomic DNA sequences: An application of dynamic programming and neural networks. *Nucl. Acids Res.* 21:607-613.

Solovyev, V.V., Salamov, A.A., and Lawrence, C.B. 1994. Predicting internal exons by oligonucleotide composition and discriminant analysis of spliceable open reading frames. *Nucl. Acids Res.* 22:5156-5163.

Staden, R. 1984. Measurements of the effects that coding for a protein has on a DNA sequence and their use for finding genes. *Nucl. Acids Res.* 12:505-519.

Uberbacher, E.C. and Mural, R.J. 1991. Locating protein-coding regions in human DNA sequences by a multiple sensor-neural network approach. *Proc. Natl. Acad. Sci. U.S.A.* 88:11261-11265.

Xu, Y., Mural, R., Shah, M., and Uberbacher, E. 1994a. Recognizing exons in genomic sequence using GRAIL II. *In* Genetic Engineering, Principles and Methods (J.K. Setlow, ed.) vol. 15, pp. 241-253. Plenum, New York.

Xu, Y., Mural, R.J., and Uberbacher, E.C. 1994b. Constructing gene models from accurately-predicted exons: An application of dynamic programming. *CABIOS* 10:613-623.

Key References

Altschul, S.F., Gish, W., Miller, W., Myers, E.W., and Lipman, D.J. 1990. Basic local alignment search tool. *J. Mol. Biol.* 215:403-410.

Antequera, F. and Bird, A. 1993. Number of CpG islands and genes in human and mouse. *Proc. Natl. Acad. Sci. U.S.A.* 90:11995-11999.

Bairoch, A. 1993. The PROSITE dictionary of sites and patterns in proteins, its current status. *Nucl. Acids Res.* 21:3097-3103.

Bairoch, A. and Boeckman, B. 1993. The SWISS-PROT protein sequence data bank, recent developments. *Nucl. Acids Res.* 21:3093-3094.

Beck, S., Kelly, A., Radley, E., Khurshid, F., Alderton, R.P., and Trowsdale, J. 1992. DNA sequence analysis of 66 kb of the human MHC class II region encoding a cluster of genes for antigen processing. *J. Mol. Biol.* 228:433-441.

Benson, D., Lipman, D.J., and Ostell, J. 1993. GenBank. *Nucl. Acids Res.* 21:2963-2965.

Bilofsky, H.S. and Burks, C. 1988. The GenBank genetic sequence data bank. *Nucl. Acids Res.* 16:1861-1864.

Boguski, M.S., Lowe, T.M., and Tolstoshev, C.M. 1993. dbEST—database for "expressed sequence tags." *Nature Genet.* 4:332-333.

Brody, L.C., Abel, K.J., Castilla, L.H., Couch, F.J., McKinley, D.R., Yin, G.Y., Ho, P.P., Merajver, S., Chandrasekharappa, S.C., Xu, J., Cole, J.L., Struewing, J.P., Valdes, J.M., Collins, F.S., and Weber, B.L. 1995. Construction of a transcription map surrounding the BRCA1 locus of human chromosome 17. *Genomics* 26:238-247.

Fields, C., Adams, M.D., White, O., and Venter, J.C. 1994. How many genes in the human genome? *Nature Genet.* 7:345-346.

Gardiner-Garden, M. and Frommer, M. 1987. CpG islands in vertebrate genomes. *J. Mol. Biol.* 196:261-282.

John, R.M., Robbins, C.A., and Myers, R.M. 1994. Identification of genes within CpG-enriched DNA from human chromosome 4p16.3. *Human Mol. Gen.* 3:1611-1616.

Jurka, J., Walichiewicz, J., and Milosavljevic, A. 1992. Prototypic sequences from human repetitive DNA. *J. Mol. Evol.* 35:286-291.

Larsen, F., Gundersen, G., Lopez, R., and Prydz, H. 1992. CpG islands as gene markers in the human genome. *Genomics* 13:1095-1107.

Lawrence, B.J., Schwabe, W., Kloschis, P., Coy, J.F., Poustka, A., Brennan, M.B., and Hochgeschwender, U. 1994. Rapid identification of gene sequences for transcriptional map assembly by direct cDNA screening of genomic reference libraries. *Hum. Mol. Gen.* 3:2014-2023.

Marshall, E. 1995. A strategy for sequencing the genome 5 years early. *Science* 267:783-784.

Pearson, W.R. and Lipman, D.J. 1988. Improved tools for biological sequence comparison. *Proc. Natl. Acad. Sci. U.S.A.* 85:2444-2448.

Peltoketo, H., Isomaa, V., Maeentausta, O., and Vihko, R. 1988. Complete amino acid sequence of human placenta 17-β-hydroxysteroid dehydrogenase deduced from cDNA. *FEBS Lett.* 239:73-77.

Smith, T.F. and Waterman, M.S. 1981. Identification of common molecular subsequences. *J. Mol. Biol.* 147:195-197.

Smith, M.W., Holmsen, A.L., Wei, Y.H., Peterson, M., and Evans, G.A. 1994. Genomic sequence sampling: A strategy for high resolution sequence-based physical mapping of complex genomes. *Nature Genet.* 6:40-47.

Wiginton, D.A., Kaplan, D., States, J.C., Akeson, A.L., Perme, C.M., Bilyk, I.J., Vaughn, A.J., Lattier, D.C., and Hutton, J.J. 1986. Complete sequence and structure of the gene for human adenosine deaminase. *Biochemistry* 25:8234-8244.

Xu, H., Wei, H., Tassone, F., Graw, F., Gardiner, K., and Weissman, S. 1995a. Search for genes from the dark band region of chromosome 21. *Genomics* 27:1-8.

Xu, Y., Mural, R.J., and Uberbacher, E.C. 1995b. Correcting sequencing errors in DNA coding regions using a dynamic programming approach. *CABIOS* 11:117-124.

Contributed by Edward C. Uberbacher,
Doug Hyatt, and Manesh Shah
Oak Ridge National Laboratory
Oak Ridge, Tennessee

Using RepeatMasker to Identify Repetitive Elements in Genomic Sequences

Maja Tarailo-Graovac[1] and Nansheng Chen[1]

[1]Simon Fraser University, Burnaby, British Columbia, Canada

ABSTRACT

RepeatMasker is a popular software tool widely used in computational genomics to identify, classify, and mask repetitive elements, including low-complexity sequences and interspersed repeats. RepeatMasker searches for repetitive sequence by aligning the input genome sequence against a library of known repeats, such as Repbase. Here, we describe two Basic Protocols that provide detailed guidelines on how to use RepeatMasker, either via the Web interface or command-line Unix/Linux system, to analyze repetitive elements in genomic sequences. Sequence comparisons in RepeatMasker are usually performed by the alignment program cross_match, which requires significant processing time for larger sequences. An Alternate Protocol describes how to reduce the processing time using an alternative alignment program, such as WU-BLAST. Further, the advantages, limitations, and known bugs of the software are discussed. Finally, guidelines for understanding the results are provided. *Curr. Protoc. Bioinform.* 25:4.10.1-4.10.14. © 2009 by John Wiley & Sons, Inc.

Keywords: RepeatMasker • genome annotation • repetitive elements • repeat library • cross_match • WU-BLAST • RECON

INTRODUCTION

RepeatMasker (developed by A.F.A. Smit, R. Hubley, and P. Green; see *http://www.repeatmasker.org/*) was designed to identify and annotate repetitive elements in nucleotide sequences and mask them for further analysis. The repetitive elements, including low-complexity DNA sequences and interspersed repeats, are annotated and replaced by Ns, Xs, or lowercase letters (see below for options) in the corresponding positions of the DNA sequence. The new addition to the RepeatMasker package is a program that also identifies repetitive elements within protein sequences. Here, we focus on utilizing RepeatMasker to identify repetitive elements in genomic sequences. To run RepeatMasker, one needs to select the repeat library files, which contain repetitive elements consensus sequences. Currently, Repbase Update (Jurka, 2001; Jurka et al. 2005; *http://www.girinst.org/*) is the largest commercially available repeat library (free for academic use) and covers a number of organisms including human, rodent, zebrafish, *Drosophila*, and *Arabidopsis thaliana*. Library files for organisms that do not have Repbase Update library files can be generated ab initio using RECON (Bao and Eddy, 2002; *http://selab.janelia.org/recon.html*) or RepeatScout (*http://bix.ucsd.edu/repeatscout/*; Price et al., 2005). The newest version of RECON, v. 1.06, was released recently and is available from the RepeatModeler package at *http://www.repeatmasker.org/RepeatModeler.html*. Sequence comparisons in RepeatMasker are usually carried out by the program cross_match, developed by Phil Green (*http://www.phrap.org/consed/consed.html#howToGet*). One can also use WU-BLAST (*http://info.cchmc.org/help/wublast.html*; see Alternate Protocol) to replace cross_match for fast processing.

Current Protocols in Bioinformatics 4.10.1-4.10.14, March 2009
Published online March 2009 in Wiley Interscience (www.interscience.wiley.com).
DOI: 10.1002/0471250953.bi0410s25
Copyright © 2009 John Wiley & Sons, Inc.

BASIC PROTOCOL 1

USING RepeatMasker VIA THE WEB INTERFACE

RepeatMasker may be accessed through the Web at *http://www.repeatmasker.org/cgi-bin/WEBRepeatMasker*. Unlike the command-line version of RepeatMasker (see Basic Protocol 2), Web RepeatMasker has a nucleotide sequence size limit of 100 kb. The attempt to analyze a sequence larger than 100 kb fails (whereupon a prompt is displayed in a message window, shown in Fig. 4.10.1). Sequences shorter than 100 kb are readily analyzed using the Web RepeatMasker, with the time needed for processing correlating with the length of the sequence. For faster service outside North America, there are RepeatMasker mirror sites in Germany, Israel, and Australia.

On the other hand, if one routinely submits large sequences for analysis, it may be better to download the command-line version and run RepeatMasker locally (see Basic Protocol 2). Importantly, if the query sequence exceeds the 100-kb limit, the only choice is to download RepeatMasker and run it locally.

Necessary Resources

Hardware

Any Internet-connected computer

Software

Web browser: e.g., Mozilla Firefox or Internet Explorer

Files

A FASTA file (*APPENDIX 1B*) or a collection of FASTA files can be processed via the Web interface. Note that the size limit is 100 kb for RepeatMasker via Web. The example file used in this protocol is a 22,539-bp human genomic DNA sequence from the UCSC Genome Browser (*http://genome.ucsc.edu/cgi-bin/hgGateway*). The coordinate is chr10:62743355-62765893.

1. Point the Web browser to *http://www.repeatmasker.org/cgi-bin/WEBRepeatMasker*. Load the FASTA sequence file (maximum 100 kb) by entering the sequence name or browsing the file. Alternatively, paste the FASTA sequence (maximum 100 kb) into the indicated text field.

   RepeatMasker will return an error message if the input sequence contains non-DNA symbols or if the sequence is too long.

2. Select a format for results from the two radio buttons next to "return format": "html" or "tar file."

   If "html" is selected, the results will be written as an html file. If "tar file" is selected, the results will be packed into an archive using the Unix "tar" protocol. For the example here, select "html."

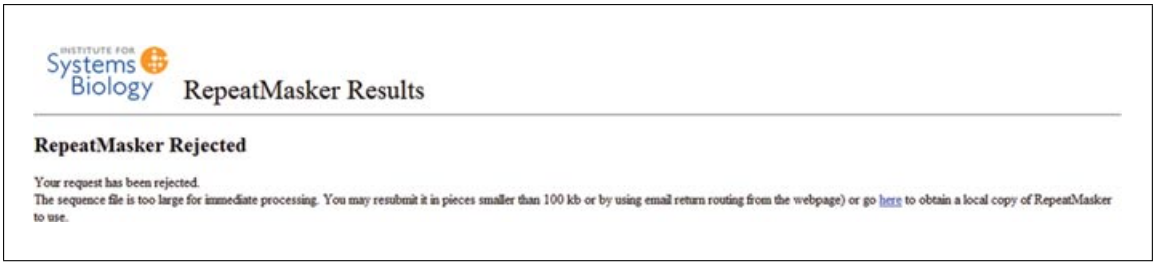
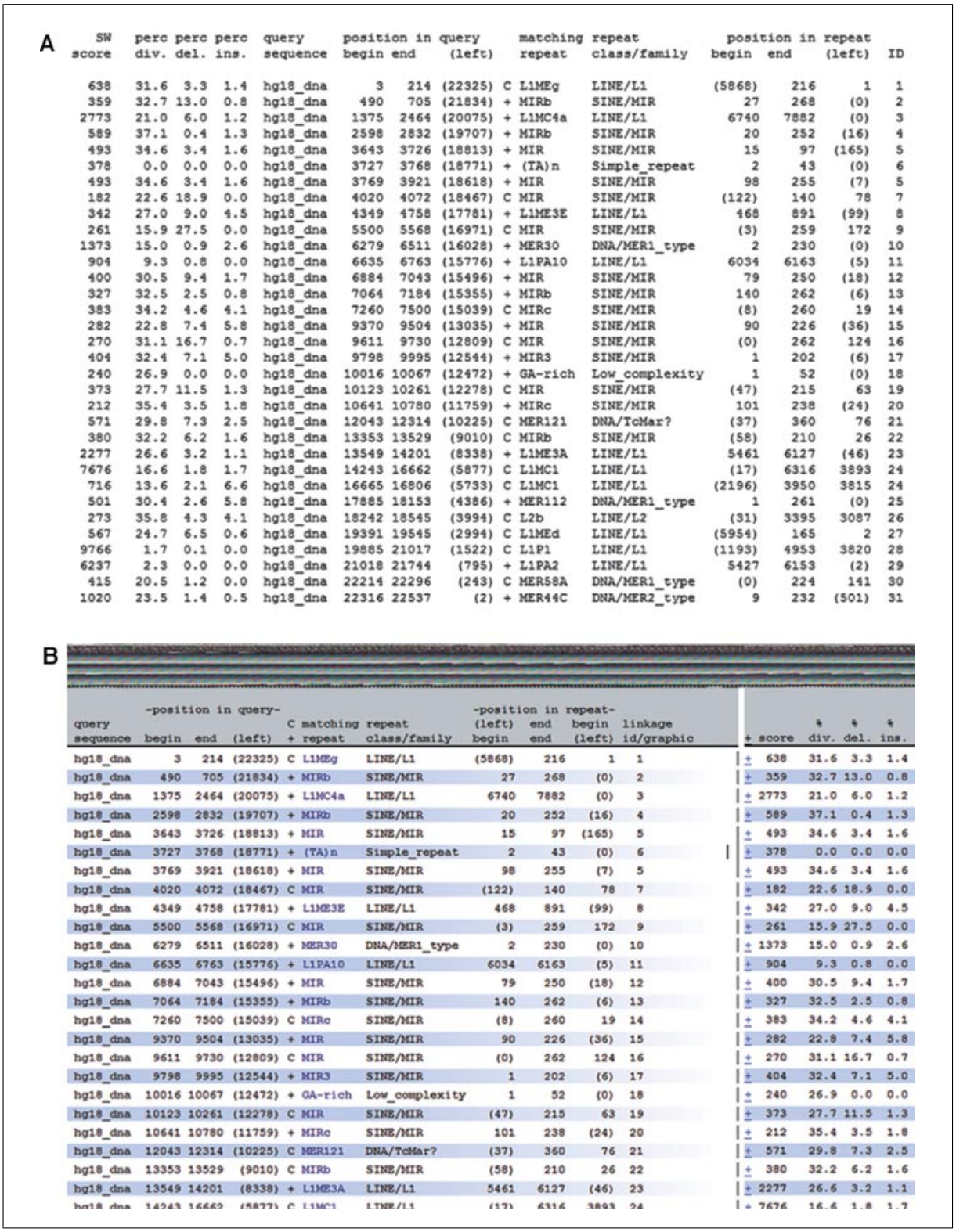

Systems Biology
RepeatMasker Results
RepeatMasker Rejected
Your request has been rejected.
The sequence file is too large for immediate processing. You may resubmit it in pieces smaller than 100 kb or by using email return routing from the webpage) or go here to obtain a local copy of RepeatMasker to use.

Figure 4.10.1 Sequences with length >100 kb cannot be processed via the Web interface; user is informed by the RepeatMasker to consider alternate methods.



A

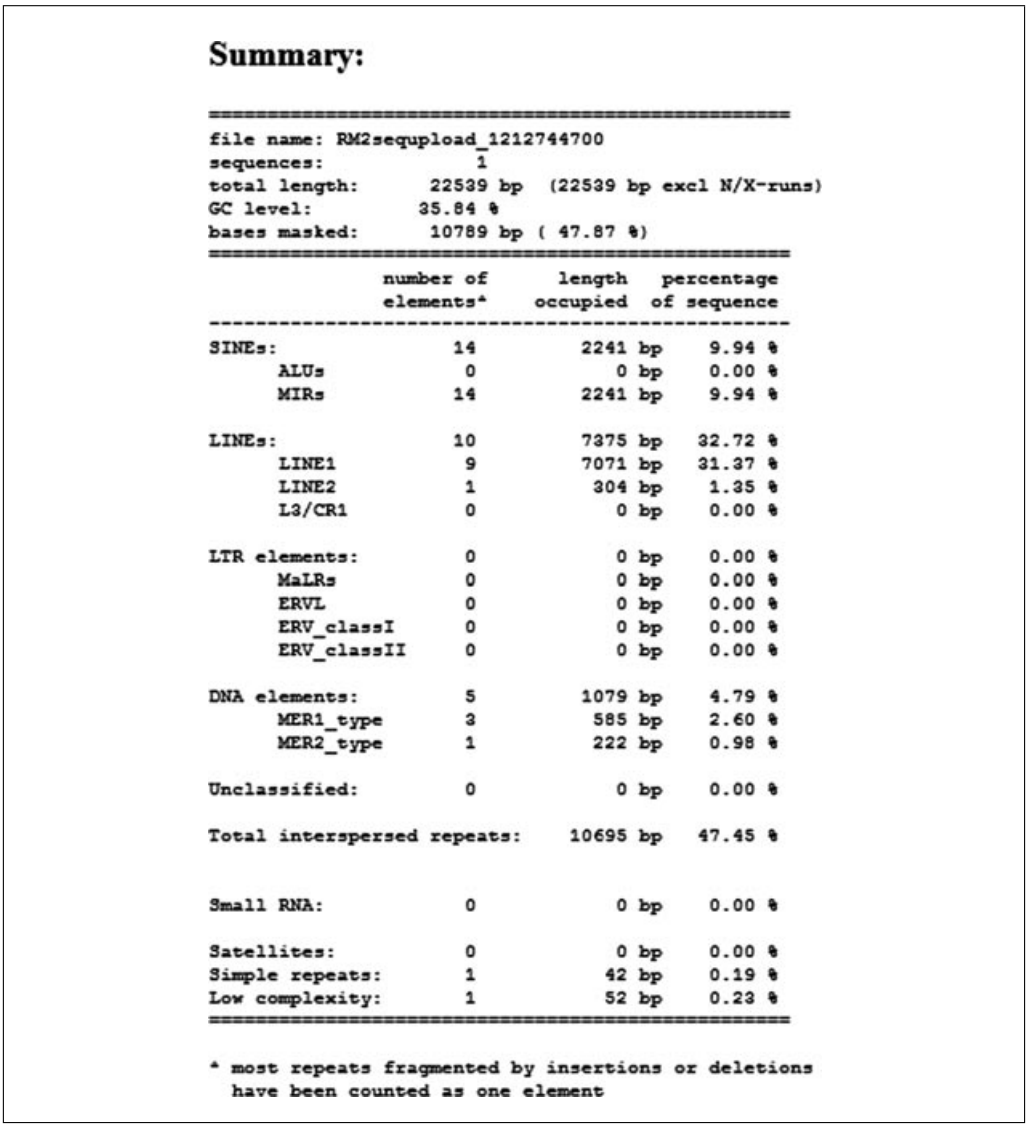
SW score	perc div.	perc del.	perc ins.	query sequence	position in query begin	position in query end	position in query (left)		matching repeat	repeat class/family	position in repeat begin	position in repeat end	position in repeat (left)	ID
638	31.6	3.3	1.4	hg18_dna	3	214	(22325)	C	L1MEg	LINE/L1	(5868)	216	1	1
359	32.7	13.0	0.8	hg18_dna	490	705	(21834)	+	MIRb	SINE/MIR	27	268	(0)	2
2773	21.0	6.0	1.2	hg18_dna	1375	2464	(20075)	+	L1MC4a	LINE/L1	6740	7882	(0)	3
589	37.1	0.4	1.3	hg18_dna	2598	2832	(19707)	+	MIRb	SINE/MIR	20	252	(16)	4
493	34.6	3.4	1.6	hg18_dna	3643	3726	(18813)	+	MIR	SINE/MIR	15	97	(165)	5
378	0.0	0.0	0.0	hg18_dna	3727	3768	(18771)	+	(TA)n	Simple_repeat	2	43	(0)	6
493	34.6	3.4	1.6	hg18_dna	3769	3921	(18618)	+	MIR	SINE/MIR	98	255	(7)	5
182	22.6	18.9	0.0	hg18_dna	4020	4072	(18467)	C	MIR	SINE/MIR	(122)	140	78	7
342	27.0	9.0	4.5	hg18_dna	4349	4758	(17781)	+	L1ME3E	LINE/L1	468	891	(99)	8
261	15.9	27.5	0.0	hg18_dna	5500	5568	(16971)	C	MIR	SINE/MIR	(3)	259	172	9
1373	15.0	0.9	2.6	hg18_dna	6279	6511	(16028)	+	MER30	DNA/MER1_type	2	230	(0)	10
904	9.3	0.8	0.0	hg18_dna	6635	6763	(15776)	+	L1PA10	LINE/L1	6034	6163	(5)	11
400	30.5	9.4	1.7	hg18_dna	6884	7043	(15496)	+	MIR	SINE/MIR	79	250	(18)	12
327	32.5	2.5	0.8	hg18_dna	7064	7184	(15355)	+	MIRb	SINE/MIR	140	262	(6)	13
383	34.2	4.6	4.1	hg18_dna	7260	7500	(15039)	C	MIRc	SINE/MIR	(8)	260	19	14
282	22.8	7.4	5.8	hg18_dna	9370	9504	(13035)	+	MIR	SINE/MIR	90	226	(36)	15
270	31.1	16.7	0.7	hg18_dna	9611	9730	(12809)	C	MIR	SINE/MIR	(0)	262	124	16
404	32.4	7.1	5.0	hg18_dna	9798	9995	(12544)	+	MIR3	SINE/MIR	1	202	(6)	17
240	26.9	0.0	0.0	hg18_dna	10016	10067	(12472)	+	GA-rich	Low_complexity	1	52	(0)	18
373	27.7	11.5	1.3	hg18_dna	10123	10261	(12278)	C	MIR	SINE/MIR	(47)	215	63	19
212	35.4	3.5	1.8	hg18_dna	10641	10780	(11759)	+	MIRc	SINE/MIR	101	238	(24)	20
571	29.8	7.3	2.5	hg18_dna	12043	12314	(10225)	C	MER121	DNA/TcMar?	(37)	360	76	21
380	32.2	6.2	1.6	hg18_dna	13353	13529	(9010)	C	MIRb	SINE/MIR	(58)	210	26	22
2277	26.6	3.2	1.1	hg18_dna	13549	14201	(8338)	+	L1ME3A	LINE/L1	5461	6127	(46)	23
7676	16.6	1.8	1.7	hg18_dna	14243	16662	(5877)	C	L1MC1	LINE/L1	(17)	6316	3893	24
716	13.6	2.1	6.6	hg18_dna	16665	16806	(5733)	C	L1MC1	LINE/L1	(2196)	3950	3815	24
501	30.4	2.6	5.8	hg18_dna	17885	18153	(4386)	+	MER112	DNA/MER1_type	1	261	(0)	25
273	35.8	4.3	4.1	hg18_dna	18242	18545	(3994)	C	L2b	LINE/L2	(31)	3395	3087	26
567	24.7	6.5	0.6	hg18_dna	19391	19545	(2994)	C	L1MEd	LINE/L1	(5954)	165	2	27
9766	1.7	0.1	0.0	hg18_dna	19885	21017	(1522)	C	L1P1	LINE/L1	(1193)	4953	3820	28
6237	2.3	0.0	0.0	hg18_dna	21018	21744	(795)	+	L1PA2	LINE/L1	5427	6153	(2)	29
415	20.5	1.2	0.0	hg18_dna	22214	22296	(243)	C	MER58A	DNA/MER1_type	(0)	224	141	30
1020	23.5	1.4	0.5	hg18_dna	22316	22537	(2)	+	MER44C	DNA/MER2_type	9	232	(501)	31

B

query sequence	-position in query- begin	end	(left)	C	matching + repeat	repeat class/family	-position in repeat- (left) begin	end	begin (left)	linkage id/graphic	+	score	% div.	% del.	% ins.
hg18_dna	3	214	(22325)	C	L1MEg	LINE/L1	(5868)	216	1	1	+	638	31.6	3.3	1.4
hg18_dna	490	705	(21834)	+	MIRb	SINE/MIR	27	268	(0)	2	+	359	32.7	13.0	0.8
hg18_dna	1375	2464	(20075)	+	L1MC4a	LINE/L1	6740	7882	(0)	3	+	2773	21.0	6.0	1.2
hg18_dna	2598	2832	(19707)	+	MIRb	SINE/MIR	20	252	(16)	4	+	589	37.1	0.4	1.3
hg18_dna	3643	3726	(18813)	+	MIR	SINE/MIR	15	97	(165)	5	+	493	34.6	3.4	1.6
hg18_dna	3727	3768	(18771)	+	(TA)n	Simple_repeat	2	43	(0)	6	+	378	0.0	0.0	0.0
hg18_dna	3769	3921	(18618)	+	MIR	SINE/MIR	98	255	(7)	5	+	493	34.6	3.4	1.6
hg18_dna	4020	4072	(18467)	C	MIR	SINE/MIR	(122)	140	78	7	+	182	22.6	18.9	0.0
hg18_dna	4349	4758	(17781)	+	L1ME3E	LINE/L1	468	891	(99)	8	+	342	27.0	9.0	4.5
hg18_dna	5500	5568	(16971)	C	MIR	SINE/MIR	(3)	259	172	9	+	261	15.9	27.5	0.0
hg18_dna	6279	6511	(16028)	+	MER30	DNA/MER1_type	2	230	(0)	10	+	1373	15.0	0.9	2.6
hg18_dna	6635	6763	(15776)	+	L1PA10	LINE/L1	6034	6163	(5)	11	+	904	9.3	0.8	0.0
hg18_dna	6884	7043	(15496)	+	MIR	SINE/MIR	79	250	(18)	12	+	400	30.5	9.4	1.7
hg18_dna	7064	7184	(15355)	+	MIRb	SINE/MIR	140	262	(6)	13	+	327	32.5	2.5	0.8
hg18_dna	7260	7500	(15039)	C	MIRc	SINE/MIR	(8)	260	19	14	+	383	34.2	4.6	4.1
hg18_dna	9370	9504	(13035)	+	MIR	SINE/MIR	90	226	(36)	15	+	282	22.8	7.4	5.8
hg18_dna	9611	9730	(12809)	C	MIR	SINE/MIR	(0)	262	124	16	+	270	31.1	16.7	0.7
hg18_dna	9798	9995	(12544)	+	MIR3	SINE/MIR	1	202	(6)	17	+	404	32.4	7.1	5.0
hg18_dna	10016	10067	(12472)	+	GA-rich	Low_complexity	1	52	(0)	18	+	240	26.9	0.0	0.0
hg18_dna	10123	10261	(12278)	C	MIR	SINE/MIR	(47)	215	63	19	+	373	27.7	11.5	1.3
hg18_dna	10641	10780	(11759)	+	MIRc	SINE/MIR	101	238	(24)	20	+	212	35.4	3.5	1.8
hg18_dna	12043	12314	(10225)	C	MER121	DNA/TcMar?	(37)	360	76	21	+	571	29.8	7.3	2.5
hg18_dna	13353	13529	(9010)	C	MIRb	SINE/MIR	(58)	210	26	22	+	380	32.2	6.2	1.6
hg18_dna	13549	14201	(8338)	+	L1ME3A	LINE/L1	5461	6127	(46)	23	+	2277	26.6	3.2	1.1
hg18_dna	14243	16662	(5877)	C	L1MC1	LINE/L1	(17)	6316	3893	24	+	7676	16.6	1.8	1.7

Figure 4.10.2 Web RepeatMasker result from an example run showing the repetitive elements annotations section, which lists cross_match summary lines; this result is available in Text File Format (**A**) and XHTML format (**B**). See Guidelines for Understanding Results and Table 4.10.1 for explanation.

>hg18_dna range=chr10:62743355-62765893 5'pad=0 3'pad=0 strand=+ repeatMasking=none
CTNNN



Summary:

```
==================================================
file name: RM2sequpload_1212744700
sequences:             1
total length:      22539 bp  (22539 bp excl N/X-runs)
GC level:         35.84 %
bases masked:      10789 bp ( 47.87 %)
==================================================
                number of      length   percentage
                elements*    occupied  of sequence
--------------------------------------------------
SINEs:               14         2241 bp    9.94 %
      ALUs            0            0 bp    0.00 %
      MIRs           14         2241 bp    9.94 %

LINEs:               10         7375 bp   32.72 %
      LINE1           9         7071 bp   31.37 %
      LINE2           1          304 bp    1.35 %
      L3/CR1          0            0 bp    0.00 %

LTR elements:        0            0 bp    0.00 %
      MaLRs           0            0 bp    0.00 %
      ERVL            0            0 bp    0.00 %
      ERV_classI      0            0 bp    0.00 %
      ERV_classII     0            0 bp    0.00 %

DNA elements:        5         1079 bp    4.79 %
      MER1_type       3          585 bp    2.60 %
      MER2_type       1          222 bp    0.98 %

Unclassified:        0            0 bp    0.00 %

Total interspersed repeats:    10695 bp   47.45 %


Small RNA:           0            0 bp    0.00 %

Satellites:          0            0 bp    0.00 %
Simple repeats:      1           42 bp    0.19 %
Low complexity:      1           52 bp    0.23 %
==================================================

* most repeats fragmented by insertions or deletions
  have been counted as one element
```

Figure 4.10.4 Web RepeatMasker result from an example run showing the Summary section, which summarizes and categorizes repetitive elements found in the query DNA sequence. See Guidelines for Understanding Results for explanation.

6. Select one of the entries from the pull-down menu next to "DNA source," each of which corresponds to a different repetitive element library.

   The default is Human. For the example here, select Human because the sequence is from the human genome.

   Note that if the query sequence is from an organism that is not listed here, the command-line version of RepeatMasker must be run locally (see Basic Protocol 2), and an appropriate repeat file from Repbase Update must be used, if there is one. If working with a genome for which Repbase does not have an appropriate repeat library, RECON (Bao and Eddy, 2002; Stein et al., 2003) or RepeatScout (http://bix.ucsd.edu/repeatscout/; Price et al., 2005) can be used to establish one from scratch.

7. In the series of pull-down menus, radio buttons, and check boxes under Lineage Annotation Options, select the appropriate options.

   These options are self-explanatory. For example, if Comparison Species is selected, the lineage-specific repeats are annotated with the RepeatMasker output with respect to the selected species.

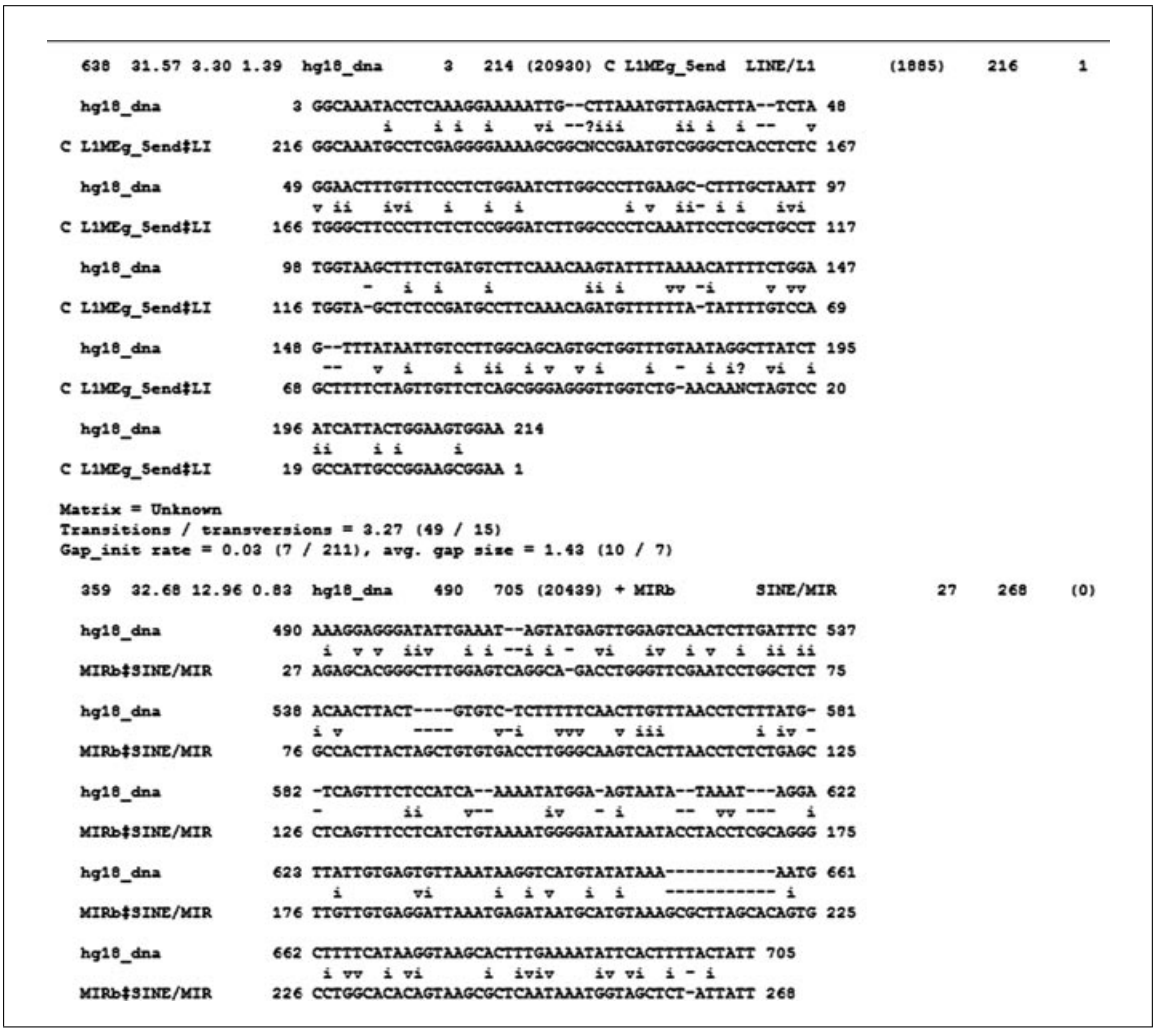

638 31.57 3.30 1.39 hg18_dna 3 214 (20930) C L1MEg_5end LINE/L1 (1885) 216 1
hg18_dna 3 GGCAAATACCTCAAAGGAAAAATTG--CTTAAATGTTAGACTTA--TCTA 48
C L1MEg_5end$LI 216 GGCAAATGCCTCGAGGGGAAAAGCGGCNCCGAATGTCGGGCTCACCTCTC 167
hg18_dna 49 GGAACTTTGTTTCCCTCTGGAAATCTTGGCCCTTGAAGC-CTTTGCTAATT 97
C L1MEg_5end$LI 166 TGGGCTTCCCTTCTCTCCGGGATCTTGGCCCCTCAAATTCCTCGCTGCCT 117
hg18_dna 98 TGGTAAGCTTTCTGATGTCTTCAAACAAGTATTTTAAAACATTTTCTGGA 147
C L1MEg_5end$LI 116 TGGTA-GCTCTCCGATGCCTTCAAACAGATGTTTTTTA-TATTTTGTCCA 69
hg18_dna 148 G--TTTATAATTGTCCTTGGCAGCAGTGCTGGTTTGTAATAGGCTTATCT 195
C L1MEg_5end$LI 68 GCTTTTCTAGTTGTTCTCAGCGGGAGGGTTGGTCTG-AACAANCTAGTCC 20
hg18_dna 196 ATCATTACTGGAAGTGGAA 214
C L1MEg_5end$LI 19 GCCATTGCCGGAAGCGGAA 1
Matrix = Unknown
Transitions / transversions = 3.27 (49 / 15)
Gap_init rate = 0.03 (7 / 211), avg. gap size = 1.43 (10 / 7)
359 32.68 12.96 0.83 hg18_dna 490 705 (20439) + MIRb SINE/MIR 27 268 (0)
hg18_dna 490 AAAGGAGGGATATTGAAAT--AGTATGAGTTGGAGTCAACTCTTGATTTC 537
MIRb$SINE/MIR 27 AGAGCACGGGCTTTGGAGTCAGGCA-GACCTGGGTTCGAATCCTGGCTCT 75
hg18_dna 538 ACAACTTACT----GTGTC-TCTTTTTCAACTTGTTTAACCTCTTTATG- 581
MIRb$SINE/MIR 76 GCCACTTACTAGCTGTGTGACCTTGGGCAAGTCACTTAACCTCTCTGAGC 125
hg18_dna 582 -TCAGTTTCTCCATCA--AAAATATGGA-AGTAATA--TAAAT---AGGA 622
MIRb$SINE/MIR 126 CTCAGTTTCCTCATCTGTAAAATGGGGATAATAATACCTACCTCGCAGGG 175
hg18_dna 623 TTATTGTGAGTGTTAAATAAGGTCATGTATATAAA----------AATG 661
MIRb$SINE/MIR 176 TTGTTGTGAGGATTAAATGAGATAATGCATGTAAAGCGCTTAGCACAGTG 225
hg18_dna 662 CTTTTCATAAGGTAAGCACTTTGAAAATATTCACTTTTACTATT 705
MIRb$SINE/MIR 226 CCTGGCACACAGTAAGCGCTCAATAAATGGTAGCTCT-ATTATT 268

Figure 4.10.5 Alignments between query sequence and consensus repetitive elements are shown if the option Show Alignments is selected.

8. In the series of pull-down menus under Advanced Options, select the appropriate Options.

   These options are straightforward as well. For example, if the user wants to make a choice between the Masking Options, users can either choose ambiguous characters, like "N" or "X" for masking, or lowercase letters, which may be more appropriate for subsequent alignments. Detailed explanation of these and additional options available can be accessed by clicking on the link to the right of each pull-down menu.

9. Click the Submit Sequence button to run RepeatMasker.

   The results displayed in the browser are shown in Figures 4.10.2, 4.10.3, 4.10.4, and 4.10.5. See Guidelines for Understanding Results for details.

BASIC PROTOCOL 2

USING THE COMMAND-LINE Unix/Linux VERSION OF RepeatMasker TO STUDY REPETITIVE ELEMENTS IN GENOMIC SEQUENCES

Using RepeatMasker to Identify Repetitive Elements in Genomic Sequences

Command-line RepeatMasker provides users with more choices and does not have the 100-kb length limit for query sequences. To run RepeatMasker locally, one must obtain RepeatMasker, cross_match, and correct repetitive libraries from Repbase Update, as detailed below. It is also possible to run RepeatMasker with WU-BLAST (see Alternate Protocol) for faster processing.

NOTE: Investigators unfamiliar with the Unix environment should read *APPENDIX 1C* and *APPENDIX 1D*.

Necessary Resources

Hardware

Any Unix or Linux workstation

Software

RepeatMasker: The software is now licensed under the Open Source License v. 2.1, and can be downloaded from *http://www.repeatmasker.org/RMDownload.html*.

cross_match: This software is part of the Phred/Phrap/Consed (*http://www.phrap.org/consed/consed.html#howToGet*; also see *UNIT 11.2*) package. It is also free for academic use. Write to Phil Green (*phg@u.washington.edu*) and include the following information in the message: (a) name; (b) an acknowledgement of agreement to observe the licensing conditions described on the above Web site (state that cross_match is desired); (c) institution/department; (d) e-mail address for all future correspondence (ideally e-mail should be received through a Unix computer running a generic mail program, since several of the programs are sent as unencoded files which may be corrupted by some mail programs). Note that it takes up to 2 weeks for a license application to be processed.

Repbase Update: This database (*http://www.girinst.org/*; Jurka, 2001) manages a large selection of repetitive element libraries, which are required for running RepeatMasker. The library is free for download by academic users, who are required to set up accounts to access the database files by filling an online form (*http://www.girinst.org/accountservices/register.php*). Commercial users should contact Jolanta Walichiewicz (*jola@girinst.org*). Once again, if one's genome of interest does not have an appropriate repeat library file in Repbase Update, one can establish one with RECON (Bao and Eddy, 2002) or RepeatScout (*http://bix.ucsd.edu/repeatscout/*; Price et al., 2005). Stein et al. (2003) used RECON to establish a repeat library file for the round worms *C. elegans* and *C. briggsae*. RECON can also be obtained as part of a Repeat Modeler package, available for download from (*http://www.repeatmasker.org/RepeatModeler.html*). Alternatively, the RepeatScout software can also be used with RepeatMasker to identify and mask repeat family sequences from newly sequenced genomes.

Files

A FASTA file (*APPENDIX 1B*) or a collection of FASTA files can be processed via the command-line RepeatMasker. Note that there is essentially no size limit for query sequences for running RepeatMasker on the command line. The example file used in this protocol is the fully sequenced whole *Caenorhabditis elegans* genome, 102,287,094 bp in length, downloaded from the WormBase (*http://www.wormbase.org*) FTP site (*ftp://ftp.wormbase.org/pub/wormbase/genomes/elegans/sequences/dna/*).

Prepare system

1. Download and install programs—RepeatMasker, Tandem Repeat Finder (TRF), cross_match, and WU-BLAST, as well as Repbase library files. RepeatMasker is a Perl script and can be put in any desired directory.

   `cross_match` *will be e-mailed to users after contacting the authors. With an account properly set up, Repbase Update will assign a user name and password to download the repetitive library files.*

   For this example, make a directory called `repeat` *in the home directory and then copy RepeatMasker, TRF, and cross_match into this directory. For this example, type:*

```
[mta57@grouse ~]$ mkdir repeat
[mta57@grouse ~]$ cd repeat
```

2. Change the permission of the programs.

   For this example, type:

```
[mta57@grouse repeat]$ chmod u+x RepeatMasker
[mta57@grouse repeat]$ chmod u+x cross_match
[mta57@grouse repeat]$ ln -s trf321.linux.exe trf
```

3. Set the correct paths by running the Configure Script.

   First, find out where Perl is installed:

```
[mta57@grouse ~]which perl
/usr/bin/perl
```

   Then, after changing to the directory `repeat` *and the directory* `RepeatMasker`*, get the current directory path using the command* `pwd`*:*

```
[mta57@grouse RepeatMasker]$ pwd
/home/mta57/repeat/RepeatMasker
```

   Then, do the same for the `TRF` *and cross_match to get the paths to the directories.*

   To configure the program use the following script:

```
[mta57@grouse repeat]$ cd RepeatMasker
[mta57@grouse RepeatMasker]$ perl ./configure
```

   Enter the required paths; for example, to write the path to the Perl interpreter, enter:

```
Enter path: /usr/bin/perl
```

   To write the path to the location where the RepeatMasker program has been installed, enter:

```
Enter path: /home/mta57/repeat/RepeatMasker
```

   For the path to the location where the TRF program can be found, enter:

```
Enter path: /home/mta57/repeat
```

   To add a search engine, enter:

```
Enter path: /home/mta57/repeat/cross_match
```

4. Place repeat libraries in the correct directory (i.e., the same directory as the script `RepeatMasker`).

   Make sure that subdirectory `Libraries` *in the* `RepeatMasker` *directory contains* `RepeatMasker.lib` *and* `RepeatMaskerLib.embl` *files.*

5. Create a new directory for input and output files.

   Note that RepeatMasker output files will be written to the same directory as the input file resides.

   For this example, type the following:

```
[mta57@grouse repeat]$ mkdir RepeatMasker_file
[mta57@grouse repeat]$ cd RepeatMasker_file
[mta57@grouse RepeatMasker_file]$
```

   Next, download or copy the FASTA file (`current.dna.fa.gz`*) containing the sequence of C. elegans genome to the directory and unpack it:*

```
[mta57@grouse RepeatMasker_file]$ gunzip current.dna.
fa.gz
```

6. To get a brief description of the command-line parameters and options, type in the program name RepeatMasker on the command line.

   For this example:

```
[mta57@grouse RepeatMasker_file]$ ../RepeatMasker/
RepeatMasker
```

   The following contents will be returned:

```
SYNOPSIS
RepeatMasker [-options] <seqfiles(s) in fasta format>
...
default settings are for masking all type of repeats
in a primate sequence.
...
```

   Choose from a number of options:

```
-q Quick search; 5-10% less sensitive, 2-5 times
faster than default
-nolow Do not mask low_complexity DNA or simple
repeats
-div [number] Mask only those repeats < x percent
diverged from consensus seq
...
-species <query species> Specify the species or
clade of the input sequence (choose only one!)
...
contamination options
...
running options
...
output options
...
```

   To get detailed help, type in:

```
[mta57@grouse RepeatMasker_file]$../RepeatMasker/
RepeatMasker -h
```

Run RepeatMasker

7. Run command-line version of RepeatMasker on the local system:

```
% /path/to/RepeatMasker -el current.dna.fa
```

   For this example, run:

```
[mta57@grouse RepeatMasker_file]$ ../RepeatMasker/
RepeatMasker -species elegans current.dna.fa
```

   Because the example sequence is from C. elegans, the `-species elegans` *command is used, so that the C. elegans Repbase repetitive element library file is used.*

   The result files will be written into the directory `RepeatMasker_file`*, the same directory where the query sequence file(s) reside(s). For this example, the result files include:*

```
current.dna.fa.masked
current.dna.fa.log
```

```
current.dna.fa.dna.cat
current.dna.fa.dna.out
current.dna.fa.dna.tbl
```

The result files are explained in Guidelines for Understanding Results, below.

8. RepeatMasker provides users with a large array of options to meet the needs appropriate for different cases. Here, only commonly used ones are covered. For more advanced options, users are encouraged to read the help file `repeatmasker.help`, which comes with the RepeatMasker program package.

   Note that the order of the command-line options is not important when entering multiple commands.

   a. `Species options` and the `-lib` flag allow users to specify a particular library file for the corresponding organism. RepeatMasker provides common name flags for some species, like `-cat` or `-dog`, but not for all. For that reason, usage of Latin names as a species option is highly recommended. Users can also provide a repeat library file, especially if the library file is not from Repbase collection, to RepeatMasker using a `-lib` flag. The default repeat library is for primate.

   To establish one's own repeat library for RepeatMasker, use the format for IDs as recommended by `repeatmasker.help`*, e.g.:*

   ```
   >repeatname#class/subclass
   ```

   or, simply

   ```
   >repeatname#class
   ```

   b. Masking options are used for determining what kind of repeats are masked. Commonly used options within this category are: `-cutoff`, `-nolow`, and `-div`. The option `-cutoff` sets cutoff score for masking repeats when using `-lib`. The default cutoff score is 225. Lower scores give more false matches. A `-nolow` flag causes RepeatMasker not to mask low-complexity DNA or simple repeats.

   The `-div` *option sets the divergence level to limit the masking and annotation to a subset of less diverged (younger) repeats.*

   c. Some options are used to control processing speed and search parameters. Options that affect processing speed are:

   `-q` quick search; 5% to 10% less sensitive, 3× to 4× faster than default
   `-qq` rush job; ∼10% less sensitive
   `-s` slow search; 0% to 5% more sensitive; 2.5× slower than default

   These flags make significant differences when the input sequences are long. If only a quick check is desired, the `-qq` *flag may be used for fast results. On the other hand, if the quality of the result is more critical, the default (with none of the above options selected), or even* `-s`*, should be used.*

   It is possible to recruit more processors for RepeatMasker by using the `-pa(rallel)` *flag, which only works when there are many input files or if the query files are big (>50 kb).*

   WU-BLAST can be used to replace cross_match if the flag `-w(ublast)` *(see Alternate Protocol) is used.*

d. Output options support the following frequently used formats (for other available options refer to `repeatmasker.help`):

`-a`	shows the alignments in a `.align` output file;
`-small`	returns complete `.masked` sequence in lower case
`-xsmall`	returns repetitive regions in lowercase (rest capitals) rather than masked
`-x`	returns repetitive regions masked with Xs rather than Ns
`-gff`	creates an additional General Feature Finding format output

Note that the `-cut` *option is not supported in the current release of RepeatMasker; however, the function may be obtained by contacting Robert Hubley (rhubley@systemsbiology.org).*

RUNNING REPEATMASKER WITH WU-BLAST

ALTERNATE PROTOCOL

Running RepeatMasker for larger sequences (e.g., whole genome for *Homo sapiens*) will take a significant amount of time. The processing time can be reduced roughly 30-fold by using WU-BLAST as the engine for RepeatMasker, to replace cross_match (Bedell et al., 2000). Although RepeatMasker with WU-BLAST has better processing time, the combination also has some limitations: (1) low-complexity repeats are not as efficiently masked as when RepeatMasker is used with cross_match; (2) some output formats are not supported; and (3) the accuracy of the results returned by the combination of RepeatMasker with WU-BLAST has not been assessed.

NOTE: Investigators unfamiliar with the Unix environment should read *APPENDIX 1C* and *APPENDIX 1D*.

Necessary Resources

Hardware

Unix or Linux workstation

Software

RepeatMasker (see Basic Protocol 2)
WU-BLAST 2.0: contact *licensing@blast.wustl.edu*
Repbase Update repeat libraries (see Basic Protocol 2)

Files

A FASTA file or a collection of FASTA files (*APPENDIX 1B*). Note that there is no size limit for running RepeatMasker with WU-BLAST on command line. The example file used in this protocol is the fully sequenced whole *C. elegans* genome, 102,287,094 bp in length, downloaded from the WormBase (*http://www.wormbase.org*) FTP site (*ftp://ftp.wormbase.org/pub/wormbase/genomes/elegans/sequences/dna/*).

1. Download and install programs—RepeatMasker, WU-BLAST, and Repeat library files. Note that until June 2004, MaskerAid (Bedell et al., 2000) was necessary for the WU-BLAST to be used with the RepeatMasker. That functionality is now implemented and does not need to be integrated separately. For this example, make a directory called `repeat` and then copy the `RepeatMasker/` directory into this directory. To do this, first change to the home directory and then make a new directory named `repeat` using `mkdir`. Use `cd` to change directory to `repeat`, as follows:

```
[mta57@grouse ~]mkdir repeat
[mta57@grouse ~]cd repeat
```

Copy `RepeatMasker/` *into this directory. Copy WU-BLAST package into this directory as well and unpack it:*

```
[mta57@grouse repeat]$ gunzip -WU_BLAST |tar xvf -
```

`wu_blast/` *directory will be seen after unpacking*

Programs within the `wu_blast/` *directory, like* `blastp`*, and* `blastx`*, are executable after unpacking.*

2. Change the permission of the programs and the directories.

   For this example:

   ```
   [mta57@grouse repeat]$ chmod u+x RepeatMasker
   [mta57@grouse repeat]$ chmod u+x wu-blast
   ```

3. Set the correct paths by running the Configure Script, as described in Basic Protocol 2.

   To add a WU-BLAST search engine, enter:

   ```
   Enter path: /home/mta57/repeat/wu-blast
   ```

4. Create a new directory for input and output files.

   RepeatMasker output files will be written to the same directory as the input file resides.

   For this example, type the following:

   ```
   [mta57@grouse repeat]$ mkdir RepeatMasker_file
   [mta57@grouse repeat]$ cd RepeatMasker_file/
   [mta57@grouse RepeatMasker_file]$
   ```

   Next, download or copy the FASTA file (`current.dna.fa.gz`*) for C. elegans genome to the directory and unpack it:*

   ```
   [mta57@grouse RepeatMasker_file]$ gunzip current.dna.
   fa.gz
   ```

5. Run program on command line using the flag `-w(ublast)`.

   For this example, run:

   ```
   [mta57@grouse RepeatMasker_file]$ ../RepeatMasker/
   RepeatMasker -w -species elegans current.dna.fa
   ```

   Here the flag `-w` *is used to indicate that WU-BLAST is used as the matching engine; the* `-species elegans` *is used to indicate that the C. elegans Repbase repetitive element library file is used, since the sequence is from C. elegans. Note that species names that contain multiple words need to be bracketed by quotation marks (e.g.,* `"caenorhabditis elegans"`*).*

   Other than the `-w` *option, which indicates that WU-BLAST is used, the command-line parameters and options are similar to those in Basic Protocol 2.*

GUIDELINES FOR UNDERSTANDING RESULTS

The output of RepeatMasker is written into five different files in the same directory where the query sequence or sequences reside. Only three files, those with `.out`, `.masked`, and `.tbl` extensions, contain results; others store processing information and are therefore not detailed here. If RepeatMasker is run via the Web server interface, the contents of these three files are written into one page (file), shown in Figures 4.10.2, 4.10.3, and 4.10.4, respectively.

Table 4.10.1 Columns of the `.out` File from Left to Right (also see Fig. 4.10.2)

Column	Content
SW score	Smith-Waterman score of the match
Perc div.	Percent substitutions in matching region compared to the consensus
Perc del.	Percent of bases opposite a gap in the query sequence (deleted bp)
Perc ins.	Percent of bases opposite a gap in the repeat consensus (inserted bp)
Query sequence	Name of query sequence
Position in query	
Begin	Starting position of match in query sequence
End	End position of match in query sequence
(Left)	Number of bases in query sequence past the end position of the current match
Matching repeat	
Repeat	Name of repeat
Class/family	The class of the repeat
Position in repeat[a]	
Begin	Starting position of match in repeat consensus sequence
End	End position of match in repeat consensus sequence
(Left)	Number of bases in repeat consensus sequence past the end of the current match
ID	Repeat identification number

[a]Note that if the repeat consensus matches the positive strand, the three subcolumns are begin, end, and (left); otherwise, the three subcolumns are (left), end, and begin.

The `.out` file (Fig. 4.10.2 in the Web example) is the annotation file that contains the cross_match summary lines. The file is basically self-explanatory. The columns of the `.out` file are described briefly in Table 4.10.1.

The matches (domains) are masked in the `.masked` file. This file can be parsed with the help of the BioPerl module (Bio::Tools::RepeatMasker, *http://www.bioperl.org*). The `.masked` file (Fig. 4.10.3) is the same as the query sequence, except that the repetitive elements are masked using Ns, Xs, or lowercase letters (if one has a `-x` or `-xsmall` flag on command line or checked the box "Mask with Xs or lower case to distinguish masked regions from Ns already in query" on the RepeatMasker Web site).

The `.tbl` file (Fig. 4.10.4) summarizes the annotation results shown in the `.out` file. Notably, the `.tbl` file states the percentage repetitive elements coverage.

COMMENTARY

Background Information

How RepeatMasker works

RepeatMasker finds and masks repetitive elements by aligning each of the query sequence(s) with each of the repeat consensus sequences in the repeat library file. Usually, cross_match is the engine that does the alignment, while RepeatMasker manages the whole process and parses the alignments. The program cross_match implements the Smith-Waterman (SW) alignment algorithm (Smith and Waterman, 1981). The program cross_match is one of the best applications for sequence alignment. The drawback of cross_match is that it is slow. To make RepeatMasker process faster, WU-BLAST can be used to replace cross_match (see Alternate Protocol). The alignment program WU-BLAST is a heuristic alignment algorithm. However, the sensitivity is reduced

when running RepeatMasker with WU-BLAST.

Critical Parameters and Troubleshooting

Limitations and known bugs

For files with multiple long sequences (e.g., a file containing whole-chromosome sequences), RepeatMasker does not work well. All of the output entries are mislabeled as the first sequence (chromosome). There is a default maximal sequence length of 4 Mb. There are two ways to work around this limitation. One way is to change the default maximal sequence length value in the RepeatMasker script. Find the following line in the script:

```
$maxsize = 4000000;
```

and modify the value. Note that the memory requirements of the program go up as this value is increased. Another way is to break down each long (>4 Mb) sequence into shorter ones.

RepeatMasker does not fail explicitly even if one's hard disk is full; it actually gives apparently normal results. Therefore, when it is noticed that the results are far from those expected, there might be a disk space problem.

Using `-q` or `-qq` (see Basic Protocol 2, step 8) can speed things up, but the sensitivity is reduced. When WU-BLAST is used, the `-s` (slow) option is preferred, since the speed with WU-BLAST is reasonably fast and the masking results are better.

Analysis on smaller sequences (<2 kb) could be less accurate.

Note that previous version(s) of RepeatMasker had a problem with overwriting the files with the same names when multiple analyses were performed on the same input files. This is no longer a problem, since RepeatMasker creates output directories for each analysis.

Literature Cited

Bao, Z. and Eddy, S.R. 2002. Automated de novo identification of repeat sequence families in sequenced genomes. *Genome Res.* 12:1269-1276.

Bedell, J.A., Korf, I., and Gish, W. 2000. MaskerAid: A performance enhancement to RepeatMasker. *Bioinformatics* 16:1040-1041.

Jurka, J. 2001. Repbase update, a database and an electronic journal of repetitive elements. *Trends Genet.* 16:418-420.

Jurka, J., Kapitonov, V.V., Pavlicek, A., Klonowski, P., Kohany, O., and Walichiewicz, J. 2005. Repbase update, a database of eukaryotic repetitive elements. *Cytogenet. Genome Res.* 110:462-467.

Price, A.L., Jones, N.C., and Pevzner, P.A. 2005. De novo identification of repeat families in large genomes. *Bioinformatics* 21:Suppl 1:i351-358.

Smith, T.F. and Waterman, M.S. 1981. Identification of common molecular subsequences. *J. Mol. Biol.* 147:195-197.

Stein, L.D., Bao, Z., Blasiar, D., Blumenthal, T., Brent, M., Chen, N., Chinwalla, A., Clarke, L., Clee, C., Coghlan, A., Coulson, A., D'Eustachio, P., Fitch, D.H.A., Fulton, L., Fulton, R., Griffiths-Jones, S., Harris, T.W., Hillier, L.W., Kamath, R., Kuwabara, P.E., Marra, M., Mardis, E., Miner, T., Minx, P., Mullikin, J.C., Plumb, R.W., Rogers, J., Schein, J., Sohrmann, M., Spieth, J., Stajich, J.E., Wei, C., Willey, D., Wilson, R., Durbin, R., and Waterston, R. 2003. The genome sequence of *Caenorhabditis briggsae*: A platform for comparative genomics. *PLoS Biol.* 1:E45.

Internet Resources

http://www.repeatmasker.org/
RepeatMasker Web server

http://www.girinst.org/
Repbase Update

http://selab.janelia.org/recon.html
RECON Web site

http://bix.ucsd.edu/repeatscout/
RepeatScout Web site

http://www.phrap.org/consed/consed.html#howToGet
cross_match Web site

http://blast.wustl.edu/
WU-BLAST Web sites

http://genome.ucsc.edu/cgi-bin/hgGateway
UCSC Genome Browser

ftp://ftp.wormbase.org/pub/wormbase/genomes/elegans/sequences/dna/
WormBaseFTP site

http://www.repeatmasker.org/RepeatModeler.html
RECON site, the newest version of RECON is available from the RepeatMasker

http://www.bioperl.org
BioPerl Web site